СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Развитие различных сфер человеческой деятельности на современном этапе невозможно без широкого применения вычислительной техники и создания информационных систем различного направления. Обработка информации в подобных системах стала самостоятельным научно-техническим направлением.

После этапа построения информационной модели начинается проектирование системы. На этом этапе производится выбор технологических решений, на основе которых будет построена информационная система.

Информация в современном мире превратилась в один из наиболее важных ресурсов, а информационные системы (ИС) стали необходимым инструментом практически во всех сферах деятельности. В реальных условиях проектирование – это поиск способа, который удовлетворяет требованиям функциональности системы средствами имеющихся технологий с учетом заданных ограничений [1].

Разнообразие задач, решаемых с помощью ИС, привело к появлению множества разнотипных систем, отличающихся принципами построения и заложенными в них правилами обработки информации.

Зачем барбершопу нужен собственный сайт? Владельцам бизнеса в этой сфере не всегда понятно, какая необходимость в создании веб-представительства. Однако сайт для барбершопа очень важен, ведь это прекрасный способ создания и поддержания имиджа организации, а также средство информирования клиентов о видах оказываемых услуг и о том, какие проводятся акции и действуют предложения.

Пользователи Интернета должны иметь возможность узнать о том, что собой представляет конкретный барбершоп: официальный сайт, цены на услуги, перечень проводимых процедур — это то, чем обязательно интересуются потенциальные клиенты. Качественный веб-ресурс — это и средство рекламы в Интернете. Кроме того, анализируя посещаемость веб-представительства, можно собрать важные сведения о востребованности разных видов услуг, о том, вызывают ли у клиентов интерес специальные предложения и акции.

Целью курсового проекта является разработка дизайна и web-приложения салона красоты.

1 ПОДГОТОВИТЕЛЬНЫЙ ЭТАП ПРИ СОЗДАНИИ ВЕБ-ПРИЛОЖЕНИЯ

1.1 Цели создания веб-Приложения

Целью дипломного проекта является разработка сайта визитки барбершопа CHIK-CHIK. Приложение должно быть реализовано в виде сайта с использованием для разработки языка программирования высокого уровня Java и фреймворка Spring.

Приложение должно запускаться посредством сборщика проектов maven или сервера tomcat и ему подобных, а также быть просматриваемым в любом современном браузере и иметь базу данных MySql.

1.2 Проектирование содержимого веб-приложения

Для проектирования приложения первым делом необходимо определить функциональность приложения и выделить основные сущности.

Из информационных сущностей можно выделить троих актеров – гостя, пользователя и администратора. Возможности гостя должны быть следующие:

- Просмотр информации о салоне
- Просмотр информации о контактах
- Просмотр информации об услугах
- Вход и регистрация на сайте

Возможности пользователя расширяют возможности гостя и имеют следующие функции:

- Онлайн регистрация на услуги
- Отмена регистрации на услуги
- Просмотр истории посещений

Возможности администратора расширяют возможности пользователя и имеют следующие функции:

- Добавление, редактирование и удаление пользователей
- Добавление, редактирование и удаление категорий услуг
- Добавление, редактирование и удаление услуг
- Добавление, редактирование и удаление заявок на услуги
- Смена статуса заявок на услуги

Итого получаем следующую UML диаграмму.

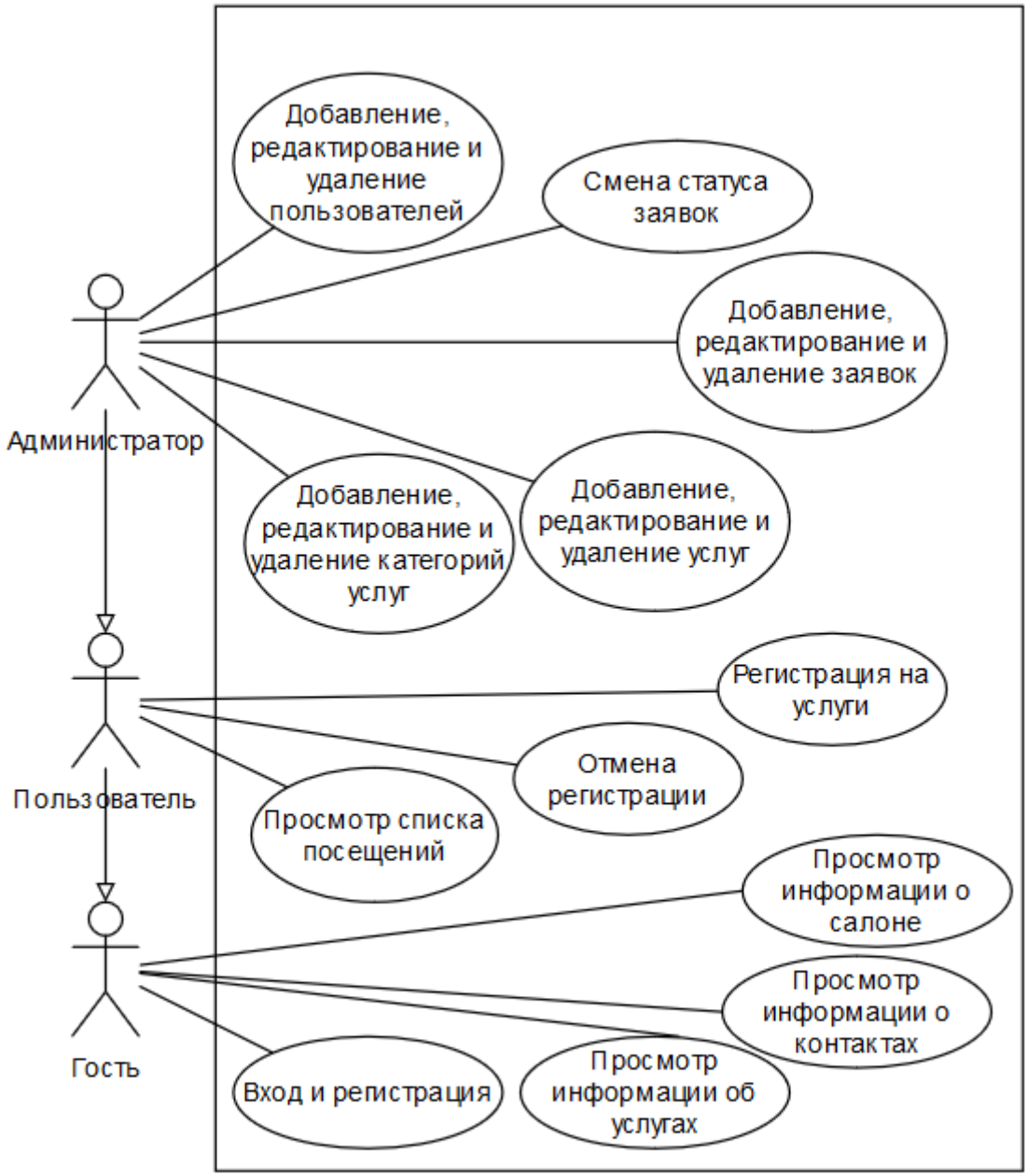

Рисунок 1.1 – Диаграмма вариантов использования

1.3 Анализ аналогов

Сайт барбершопа или же салона красоты — это визитная карточка заведения и оружие стратегического превосходства, если он создан умелыми руками. Но, прежде чем приступить к обзору уже существующих разработок, давайте разберемся, какие задачи он поможет решить.

1. Разместить информацию о салоне.

Основная функция сайта — информационная.

2. Привлечь клиентов.

Вторая основная функция сайта салона красоты (или других похожих заведений, например парикмахерской или индивидуального специалиста в области красоты) заключается в завоевании новых клиентов, в расширении целевой аудитории. Статистика показывает, что словосочетание «салон красоты» в поисковике «Яндекс» насчитывает 1184016 запросов в месяц. Слово «парикмахерская» пользователи набирают 1153975 раз за месяц. Эти данные говорят о высокой востребованности специалистов-парикмахеров и услуг бьюти-центров. Задача – перенаправить этих интересующихся людей на сайт салона красоты. Сайт салона красоты (парикмахерского салона или частного специалиста) должен привлекать новых клиентов и активно работать с целевой аудиторией. Важно суметь конвертировать эту высокую заинтересованность в количество реальных посетителей сайта [2].

3. Показать открытость. В основном люди полагают, что салон красоты предназначен для того, чтобы сделать там стрижку, покрасить волосы, пройти процедуру маникюра и педикюра. Вы можете разместить на вашем сайте салона красоты подробную информацию о том, какие услуги вы предлагаете: как стандартные, так и эксклюзивные, уникальные. Неплохо будет создать разделы, где вы укажете не только стоимость ваших услуг, но и дадите полезные советы, а также сведения по скидкам, акциям, специальным предложениям. Благодаря этому ваши клиенты смогут в онлайн-режиме выбрать те процедуры и услуги, которые им подходят. На одной из страниц можно выделить специальное место для объявлений о вакансиях, предложениях о партнерстве, ссылки на те компании. Все те процедуры, которые вы предлагаете своим клиентам (как стандартные, так и эксклюзивные), можно подробно описать на сайте.

4. Повысить доверие клиентов. Для того чтобы при помощи повысить лояльность ваших клиентов, нужно создать раздел с сертификатами и дипломами о повышении квалификации сотрудников, разместить портфолио ваших работ и отзывы потребителей. Также важно сделать доступными формы «Вопрос специалисту» и «Запись онлайн». Сайт салона красоты может быть эффективным инструментом для повышения лояльности потребителей. Для этого следует создать отдельную страницу с отзывами клиентов, с наградами и сертификатами, которыми был отмечен салон, выложить портфолио работ мастеров, предоставить посетителям возможность задать вопрос специалисту и записаться на прием к мастеру в онлайн-режиме.

5. Изучить спрос Сайт салона красоты поможет вам понять, насколько интересны и популярны ваши услуги у клиентов и посетителей. Инструменты ресурса также позволяют определить отношение пользователей к вашим скидкам, акциям. Все это можно увидеть благодаря статистике заходов, посещений, переходов.

Ниже будет представлен список сайтов салонов красоты, а также будут рассмотрены их функциональность.

Салон красоты TIFFANY [3]

Адрес сайта: https://tiffany.by/

Сайт представляет из себя лендинг (одностраничное оторажение). На нем пользователь может ознакомится с описанием сайта, перечнем предоставляемых услуг и ценой на них. Почти в каждом разделе сайта есть кнопки записи на услуги.

Сайт не предусматривает личного кабинета, а также просмотра истории посещений. При нажатии на кнопку запись, пользователю будет предложено ввести имя и номер телефона, на который позвонит оператор.

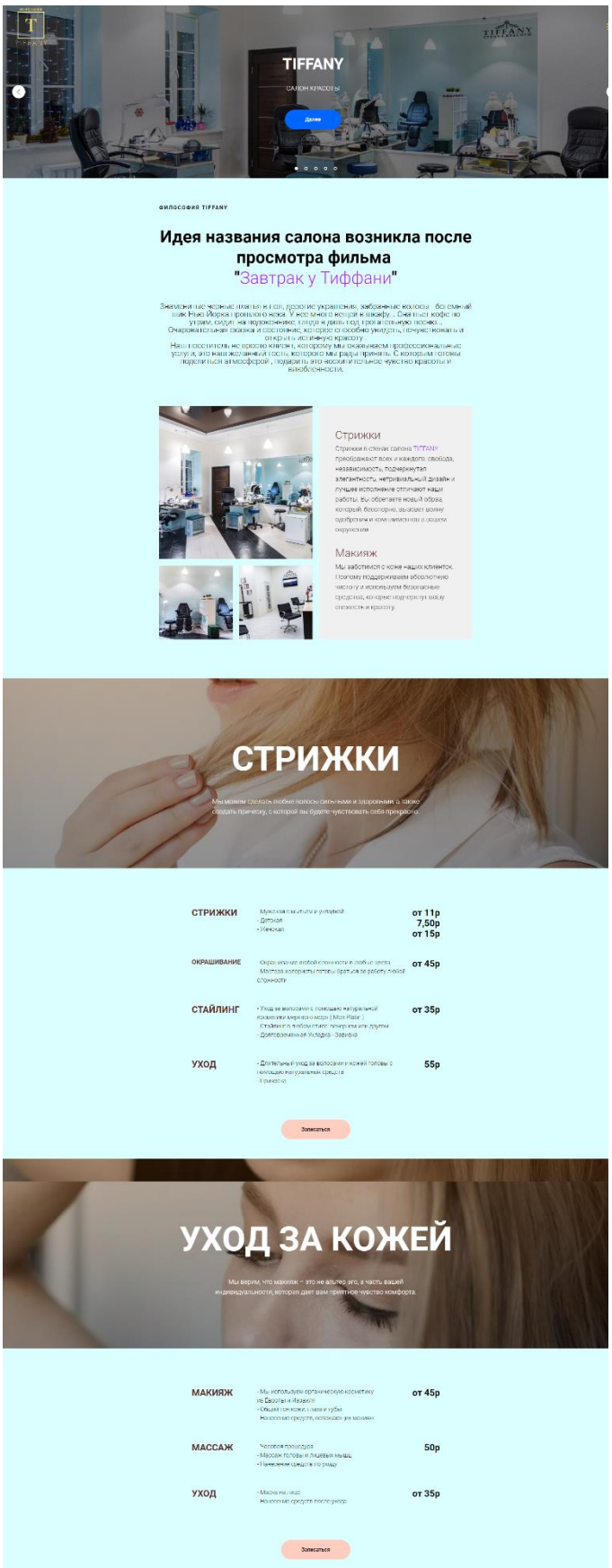

Рисунок 1.2 – Главная страница сайта салона красоты Tiffany

Студии красоты "Blond&Brown" [4]

Адрес сайта: http://blonda.by/about

Сайт содержит информацию об услугах салона красоты, такую как сами услуги, их стоимость, отзывы и контакты для связи. В контактах указаны телефоны, адрес почти и время работы студии.

Запись на сайте, как и кабинет пользователя не предусмотрены. В следившие чего, невозможно посмотреть свои прошлые записи, как и записаться на прием через сайт.

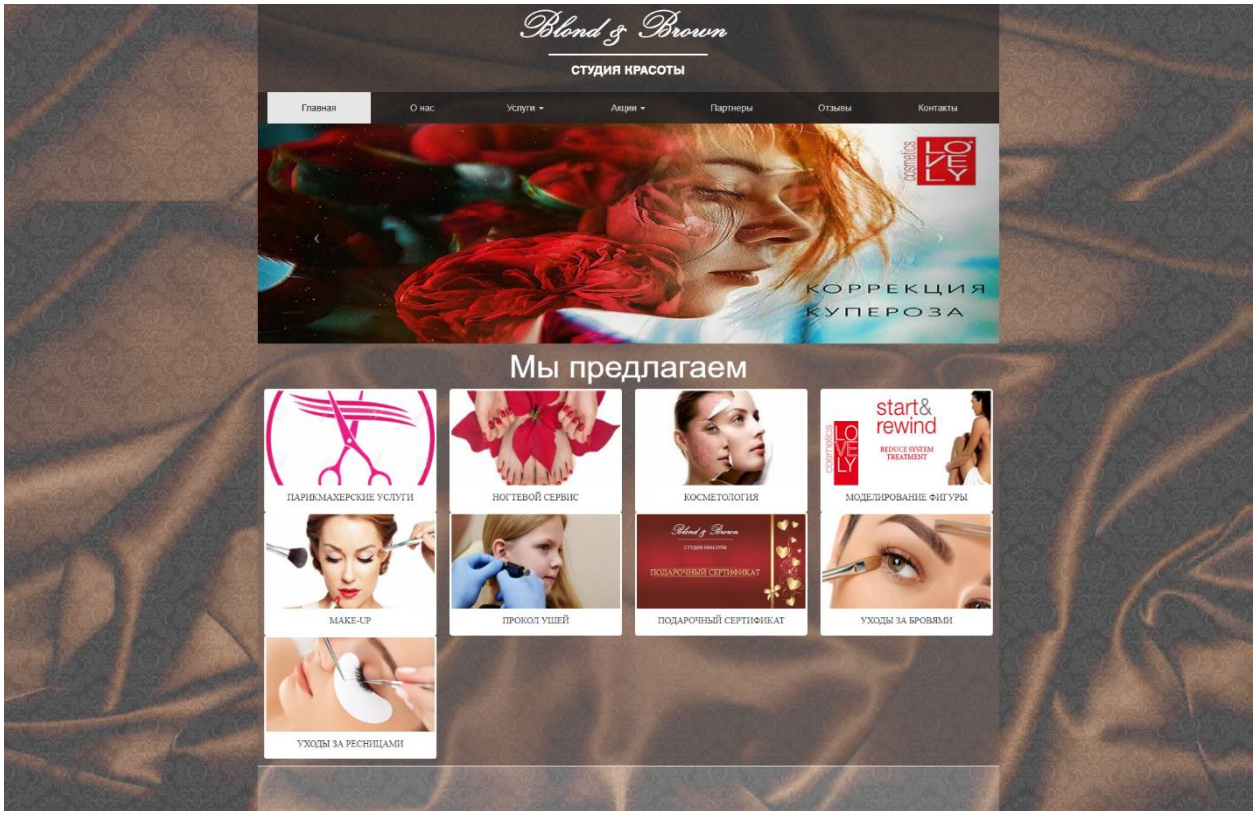

Рисунок 1.3 – Главная страница студии красоты "Blond&Brown"

Гостиница красоты «Шуры-Муры» [5]

Адрес сайта: https://shury-mury.by

Представляет из себя информационный сайт, на котором можно узнать что и по каким ценам они делают. Возможность записаться через сайт отсутствует.

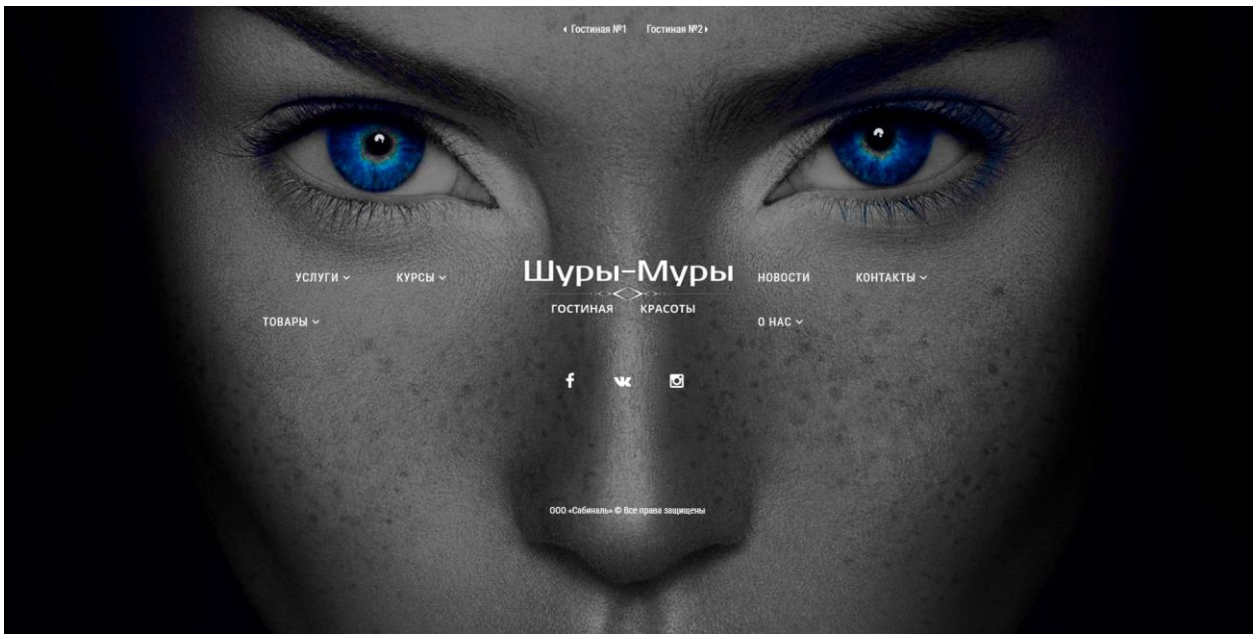

Рисунок 1.4 – Главная страница гостиницы красоты «Шуры-Муры

Студия красоты Аделины Маффей [6]

Адрес сайта: http://amstudia.by

Сайт представляет собой лендинг, на котором представлена основная информация. Список услуг не предоставлен. Тем не менее, сайт поддерживает запись онлайн. Для онлайн записи сайт использует стороннее приложение (YCLIENTS), которое производит регистрацию и запись пользователя.

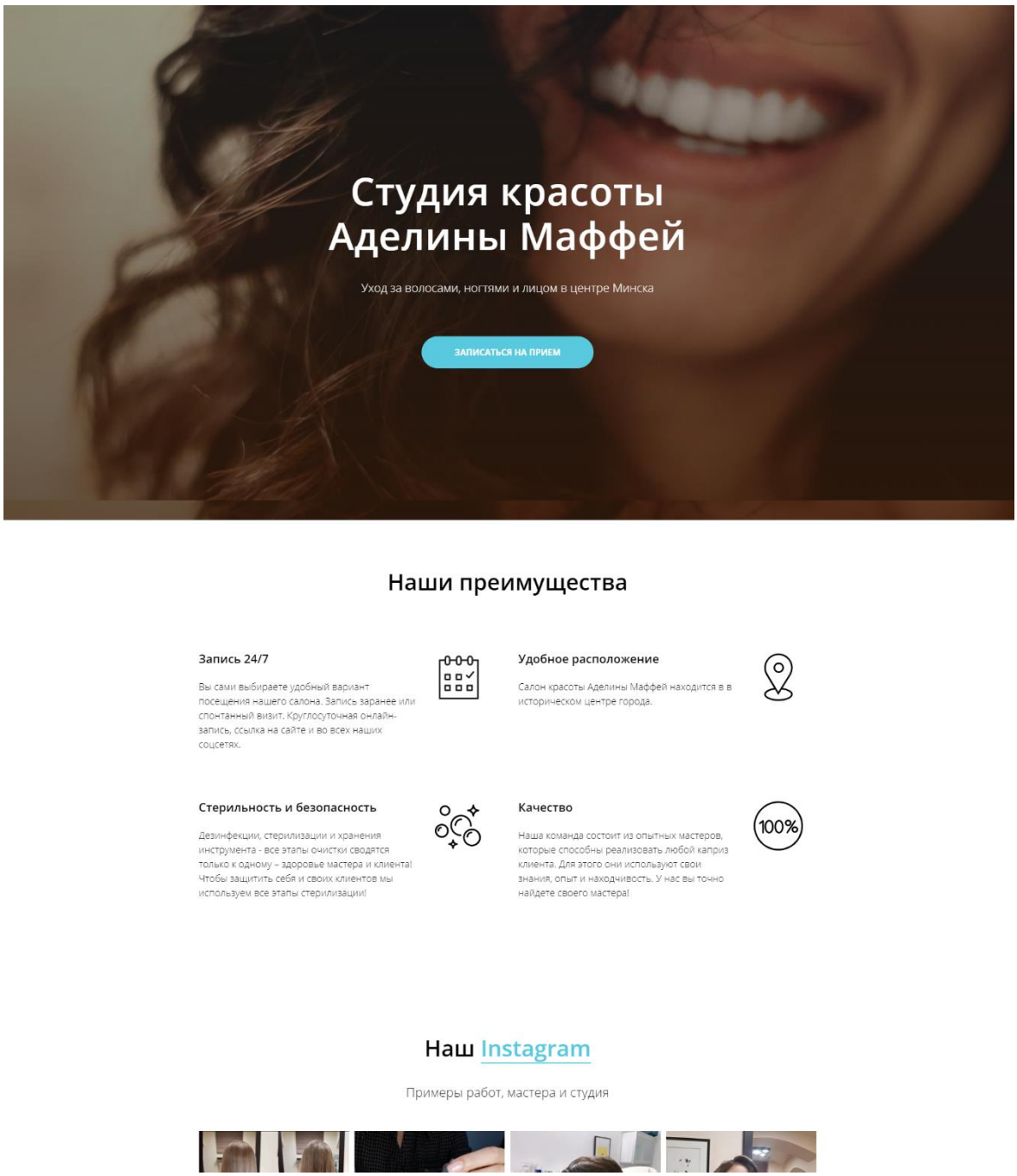

Рисунок 1.5 – Главная страница студии красоты Аделины Маффей

Студия красоты «BellaNika (Бэлла Ника)» [7]

Адрес сайта: https://bellanika.103.by

Представляет собой информационную страницу на портале. На ней можно узнать об услугах, ценах на эти услуги, а также контактные данные. Как и в примере выше, использует стороннее приложение для регистрации и записи пользователей.

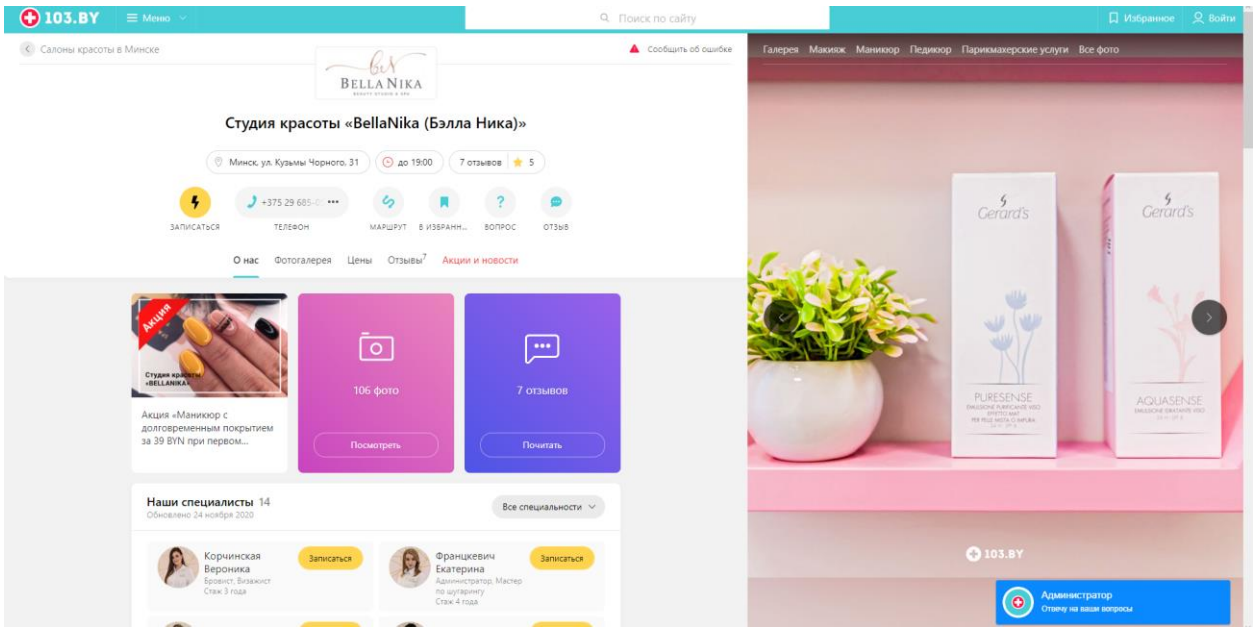

Рисунок 1.6 – Главная страница студии красоты «BellaNika (Бэлла Ника)»

Таблица 1. Сравнение сайтов салонов красоты

Салон / услуги	Информация	ЛК	История	Выбор времени услуги	Выбор времени обратной связи	База данных
TIFFANY	+	-	-	По телефону	-	Неизвестно
Blond&Brown	+	-	-	По телефону	-	Неизвестно
Шуры-Муры	+	-	-	По телефону	-	Неизвестно
Аделины Маффей	+	+	+	При записи	-	Внешняя
BellaNika	+	+	+	При записи	-	Внешняя

По приведенной таблице можно заметить, что лучшими вариантами являются два последних салона, поскольку у них есть возможность записываться через сайт, а также просматривать свои прошлые записи. Тем не менее, они не лишены недостатков – они используют внешнее приложение для управления базой данных и взаимодействия с

клиентами, что влечет за собой дополнительные расходы на содержание стороннего приложения.

Кроме того, при возникновении ошибок в внешнем приложении, вся работа будет парализована, поскольку за возвращение в строй будут отвечать не сотрудники салона.

2 РЕАЛИЗАЦИЯ

2.1 Требование к веб-приложению и программному обеспечению

Перед тем как начать разрабатывать программное средство необходимо проанализировать требования к программному средству и поставить задачи.

Цель процесса анализа требований к программному средству заключается в установлении и документировании требований к программному обеспечению.

Разработка программного средства будет вестись на операционной системе Windows 10.

При разработке программного средства будут использованы следующие технологии: язык программирования Java [8], сервер базы данных MySQL. В качестве фреймворка будет выбран фреймворк Spring Framework.

Для реализации Back-end части будут использована технология Spring Framework [9].

Для реализации Front-end части будут использованы следующие технологии: Thymeleaf [10], HTML [11], CSS [12] и JavaScript [13].

Каждая из выбранных технологий отвечает за разные аспекты работы программы.

Существует огромное количество инструментария для программирования как на стороне сервера, так и на стороне клиента, как платных, так и бесплатных.

Основные программные средства, которые будут использованы при разработке программного средства:

интегрированная среда разработки Eclipse для разработки на языках программирования Java и JavaScript, языке разметки HTML и каскадных таблиц стилей CSS с возможностями анализа кода на лету, предотвращения ошибок в коде и автоматизированными средствами рефакторинга для Java;

браузеры Opera и Google Chrome для проверки на кроссбраузерность программного средства, а также проверки результатов работы программного средства при разработке.

Все перечисленные выше серверные и клиентские технологии, а также соответствующий инструментарий идеально подходят для разработки веб-приложений.

Любой цикл разработки программного средства начинается с анализа требований. Цель этой стадии – определение детальных требований к программному средству.

Основные требования к программному средству системы управления персоналом предприятия представлены в таблице 2.

Таблица 1 - Основные требования к программному средству

Вид требования	Содержание требования
Бизнес-требование (*B*1)	Все формы программного средства должны быть удобными для пользователей
Бизнес-требование (*B*2)	Для эксплуатации интерфейса программного средства от пользователей не должно требоваться специальных технических навыков, знания технологий или программных продуктов, за исключением общих навыков работы с персональным компьютером и стандартным веб-браузером
Нефункциональное требование (*NF*1)	Программное средство должно быть адаптивно под любое устройство
Нефункциональное требование (*NF*2)	Интерфейс программного средства должен быть на русском языке
Нефункциональное требование (*NF*3)	Цветовая гамма программного средства должна иметь четыре основных цвета: белый, черный, светло-коричневый и синий.
Функциональное требование (*F*1)	В программном средстве должно быть предусмотрено разделение ролей на гостя, пользователя и администратора
Функциональное требование (*F*2)	Не авторизовавшись в системе, пользователь должен иметь доступ к навигационной панели, которая обеспечивает переход к основным пунктам меню главная, услуги, о нас, контакты, новости)
Функциональное требование (*F*3)	Графическая оболочка страниц должна делиться на следующие разделы: – навигационное меню; – ссылка «Главная»; – поле для отображения контента выбранной страницы
Функциональное требование (*F*4)	Каждому пользователю, в зависимости от роли, показывается свое меню

Требования к программному средству определены

2.2. Структура веб-приложения

Чаще всего веб-приложения состоят как минимум из трёх основных компонентов:

Клиентская часть веб приложения — это графический интерфейс. Это то, что вы видите на странице. Графический интерфейс отображается в браузере. Пользователь взаимодействует с веб-приложением именно через браузер, кликая по ссылкам и кнопкам.

Серверная часть веб-приложения — это программа или скрипт на сервере, обрабатывающая запросы пользователя (точнее, запросы браузера). Чаще всего серверная часть веб-приложения программируется на PHP. При каждом переходе пользователя по ссылке браузер отправляет запрос к серверу. Сервер обрабатывает этот запрос, вызывая некоторый PHP-скрипт, который формирует веб-страничку, описанную языком HTML, и отсылает клиенту по сети. Браузер тут же отображает полученный результат в виде очередной веб-страницы.

База данных (БД, или система управления баазми данных, СУБД) - программное обеспечение на сервере, занимающееся хранением данных и их выдачей в нужный момент. В случае форума или блога, хранимые в БД данные — это посты, комментарии, новости, и так далее. База данных располагается на сервере. Серверная часть веб-приложения (то есть, PHP скрипт) обращается к базе данных, извлекая данные, которые необходимы для формирования страницы, запрошенной пользователем.

Наше приложение также будет состоять из этих трех частей и основываться будет на Spring MVC. MVC — это не шаблон проекта, это конструкционный шаблон, который описывает способ построения структуры нашего приложения, сферы ответственности и взаимодействие каждой из частей в данной структуре.

Фреймворк Spring MVC обеспечивает архитектуру паттерна Model — View — Controller (Модель — Отображение (далее — Вид) — Контроллер) при помощи слабо связанных готовых компонентов [14]. Паттерн MVC разделяет аспекты приложения (логику ввода, бизнес-логику и логику UI), обеспечивая при этом свободную связь между ними.

- Model (Модель) инкапсулирует (объединяет) данные приложения, в целом они будут состоять из POJO («Старых добрых Java-объектов», или бинов).
- View (Отображение, Вид) отвечает за отображение данных Модели, — как правило, генерируя HTML, которые мы видим в своём браузере.
- Controller (Контроллер) обрабатывает запрос пользователя, создаёт соответствующую Модель и передаёт её для отображения в Вид.

Вся логика работы Spring MVC построена вокруг DispatcherServlet, который принимает и обрабатывает все HTTP-запросы (из UI) и ответы на них. Рабочий процесс обработки запроса DispatcherServlet'ом проиллюстрирован на следующей диаграмме:

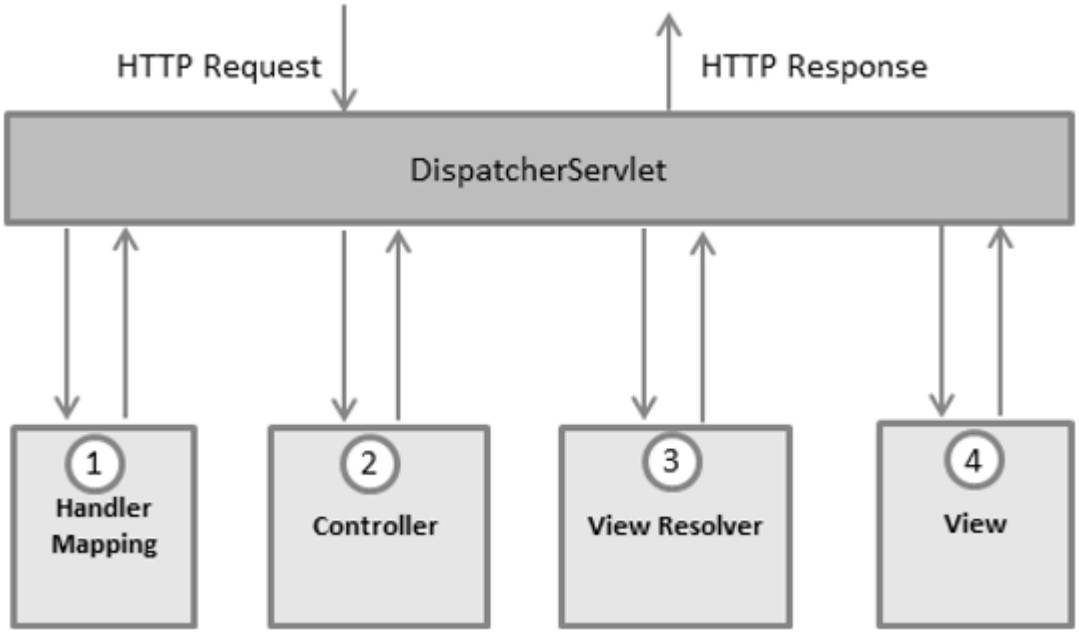

Рисунок 2.1 – Рабочий процесс обработки запроса

Ниже приведена последовательность событий, соответствующая входящему HTTP-запросу:

− После получения HTTP-запроса DispatcherServlet обращается к интерфейсу HandlerMapping, который определяет, какой Контроллер должен быть вызван, после чего, отправляет запрос в нужный Контроллер.

− Контроллер принимает запрос и вызывает соответствующий служебный метод, основанный на GET или POST. Вызванный метод определяет данные Модели, основанные на определённой бизнес-логике, и возвращает в DispatcherServlet имя Вида (View).

− При помощи интерфейса ViewResolver DispatcherServlet определяет, какой Вид нужно использовать на основании полученного имени.

− После того, как Вид (View) создан, DispatcherServlet отправляет данные Модели в виде атрибутов в Вид, который в конечном итоге отображается в браузере.

Все вышеупомянутые компоненты, а именно, HandlerMapping, Controller и ViewResolver, являются частями интерфейса WebApplicationContext extends ApplicationContext, с некоторыми дополнительными особенностями, необходимыми для создания web-приложений.

DispatcherServlet отправляет запрос контроллерам для выполнения определённых функций. Аннотация @Controllerannotation указывает, что конкретный класс является

контроллером. Аннотация @RequestMapping используется для мапинга (связывания) с URL для всего класса или для конкретного метода обработчика.

Аннотация Controller определяет класс как Контроллер Spring MVC. В первом случае, @RequestMapping указывает, что все методы в данном Контроллере относятся к URL-адресу "/hello".

Следующая аннотация @RequestMapping(method = RequestMethod.GET) используется для объявления метода printHello() как дефолтного метода для обработки HTTP-запросов GET. Вы можете определить любой другой метод как обработчик всех POST-запросов по данному URL-адресу.

Вы можете написать вышеуказанный Контроллер по-другому, указав дополнительные атрибуты для аннотации @RequestMapping следующим образом

@RequestMapping(value = "/hello", method = RequestMethod.GET)
Атрибут «value» указывает URL, с которым мы связываем данный метод (value = "/hello"), далее указывается, что этот метод будет обрабатывать GET-запросы (method = RequestMethod.GET). Также, нужно отметить важные моменты в отношении приведённого выше контроллера:

- Вы определяете бизнес-логику внутри связанного таким образом служебного метода. Из него Вы можете вызывать любые другие методы.
- Основываясь на заданной бизнес-логике, в рамках этого метода Вы создаёте Модель (Model). Вы можете добавлять аттрибуты Модели, которые будут добавлены в Вид (View). В примере выше мы создаём Модель с атрибутом «message».
- Данный служебный метод возвращает имя Вида в виде строки String. В данном случае, запрашиваемый Вид имеет имя «hello».

Spring MVC поддерживает множество типов Видов для различных технологий отображения страницы. В том числе — JSP, HTML, PDF, Excel, XML, Velocity templates, XSLT, JSON, каналы Atom и RSS, JasperReports и проч. Но чаще всего используются шаблоны JSP, написанные при помощи JSTL или HTML файлы, используя Thymeleaf.

На рисунке 3.2 представлена диаграмма последовательности, которая представляет собой один конкретный экземпляр работы программы под управлением пользователя [15].

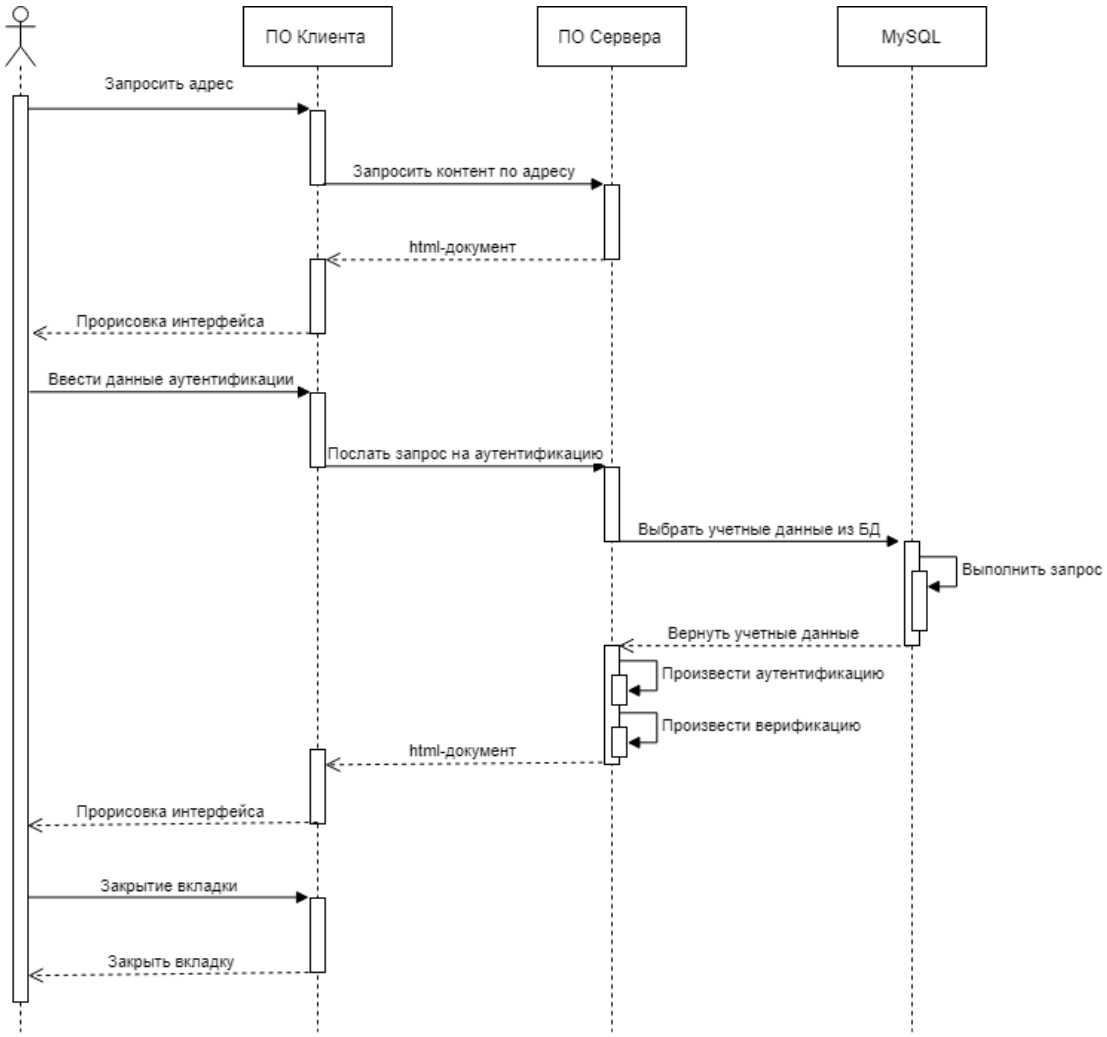

Рисунок 2.2 – Диаграмма последовательности

Диаграмма последовательности – это диаграмма, на которой для некоторого набора объектов на единой временной оси показан жизненный цикл какого-либо определённого объекта (создание – деятельность – уничтожение некой сущности) и взаимодействие актеров (действующих лиц) информационной системы в рамках какого-либо определённого прецедента (отправка запросов и получение ответов).

Основными элементами диаграммы последовательности являются обозначения объектов (прямоугольники с названиями объектов), вертикальные «линии жизни», отображающие течение времени, прямоугольники, отражающие деятельность объекта или исполнение им определенной функции (прямоугольники на пунктирной «линии жизни»), и стрелки, показывающие обмен сигналами или сообщениями между объектами.

На диаграмме последовательности изображен один актёр – пользователь системы, а также следующие объекты:

– ПО Клиента;

– ПО Сервера;

– MySQL.

Представлена последовательность действия в три этапа:

1. Пользователь вводит адрес сайта в браузере, а сервер отправляет клиенту html-документ.

2. Клиент вводит данные аутентификации. Браузер посылает запрос серверу на аутентификацию, а сервер посылает запрос на сервер MySQL, который выбирает данные и посылает их серверу. Сервер их преобразует и возвращает клиентскому браузеру. Браузер отображает информацию для клиента.

3. Клиент нажимает кнопку закрытия в браузере – вкладка закрывается. Работа системы завершена.

В этом подразделе произведено обоснование выбора технологий, а также описаны три основных типа диаграммы UML для отображения структуры данных.

2.3. Дизайн веб-приложения. Создание макета приложения

Определившись что и как должно быть, осталось спроектировать примерный пользовательский интерфейс. Пользовательский интерфейс представляет собой совокупность программных и аппаратных средств, обеспечивающих взаимодействие пользователя и вычислительной системы. Важно помнить, что интерфейсы на стадии планирования и реализации могут отличатся, поскольку при написании приложения выявляется множество проблем и изменений.

ГОСТ «Эргономика взаимодействия человек-система», введенный в 2012 г., определяет пользовательский интерфейс (ПИ) как «компоненты интерактивной системы, предоставляющие пользователю информацию и являющиеся инструментами управления для выполнения определенных задач».

Проектирование пользовательского интерфейса – это создание тестовой версии приложения. Это начальный этап разработки пользовательского интерфейса, когда распределяются функции приложения по экранам, определяются макеты экранов, содержимое, элементы управления и их поведение.

На главной странице покажем основную информацию, приведем список последних услуг. В зависимости от роли пользователя, ему будут (или не будут) показаны меню Вход, Регистрация и История.

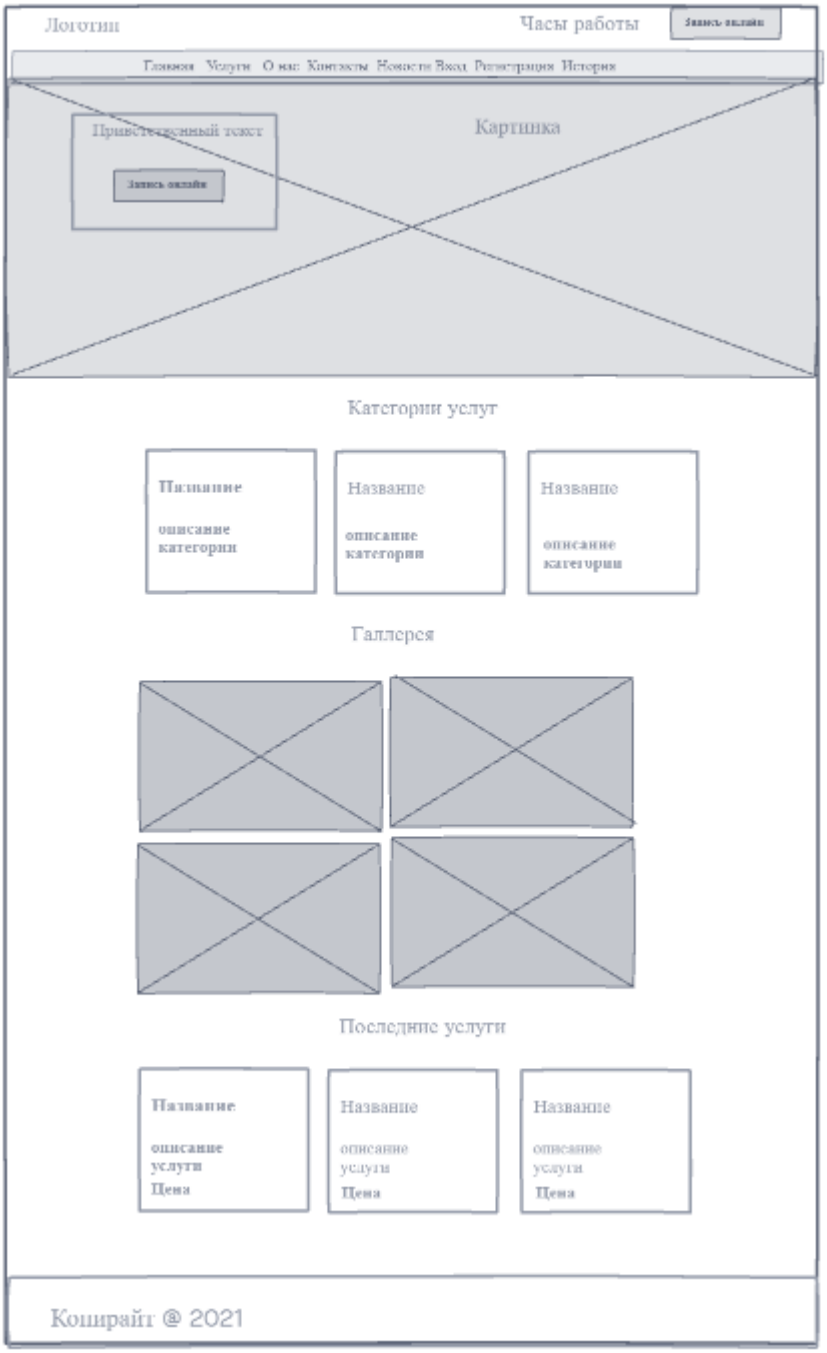

Рисунок 2.3 – Главная страница

На странице услуг отобразим аккордеонами категории услуг, а в каждом аккордеоне отобразим список услуг этой категории.

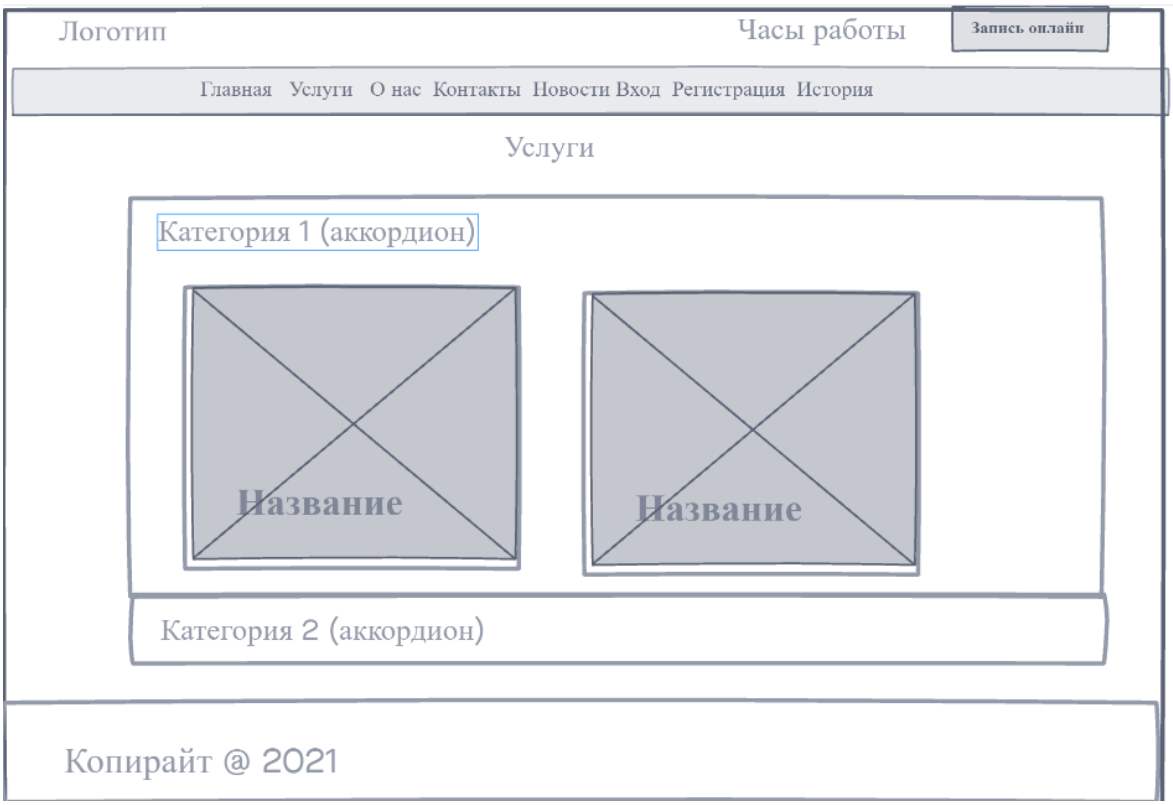

Рисунок 2.4 – Страница услуг

На странице о нас расскажем краткую информацию о салоне, а также при помощи аккордеонов приведем пару интересных фактов.

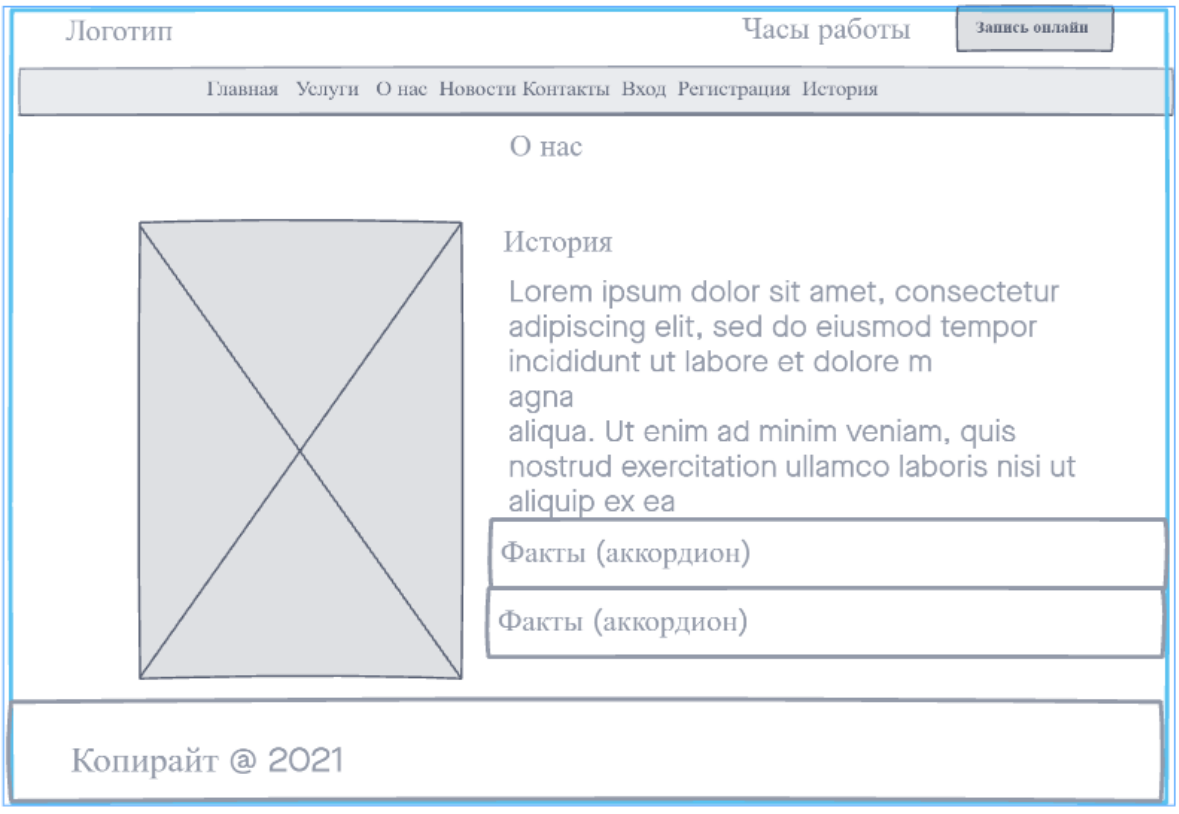

Рисунок 2.5 – Страница О Нас

Страница контактов будет содержать информацию для возможной связи с администратором.

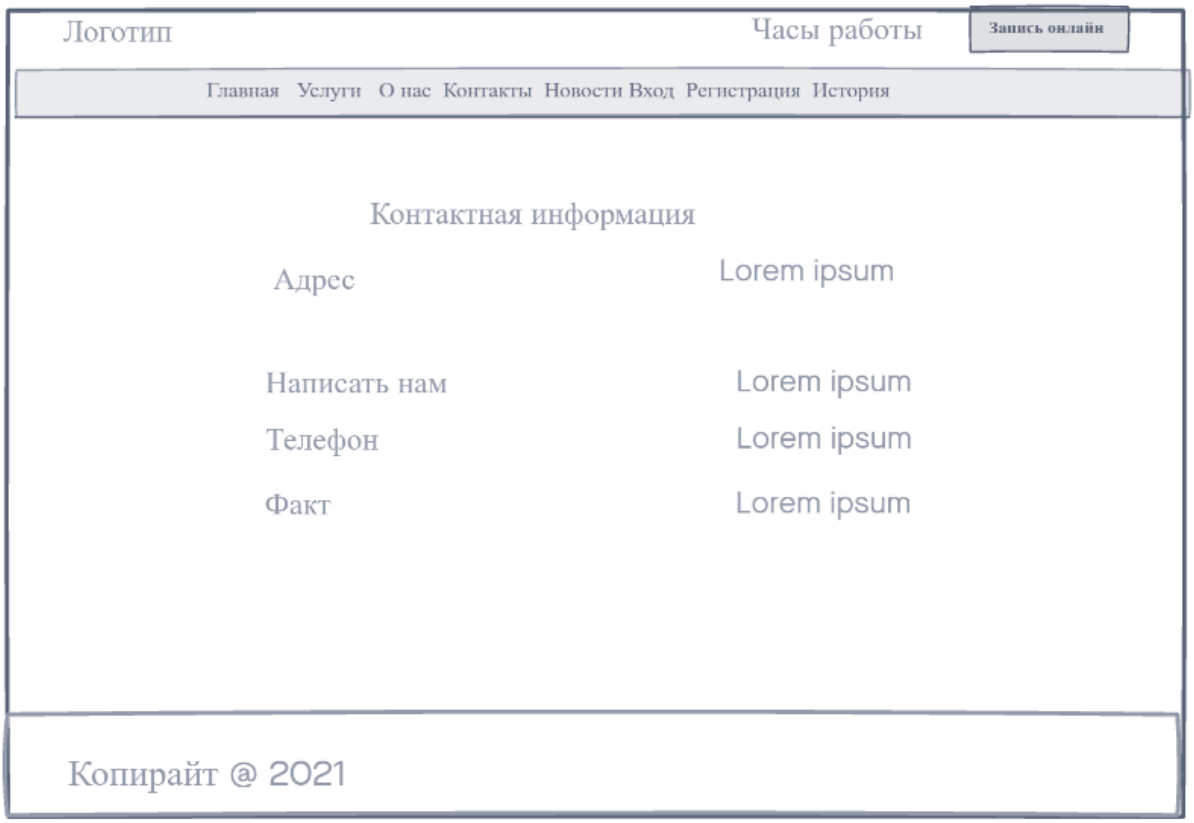

Рисунок 2.6 – Страница контакты

И страница истории, на которой авторизованному пользователю будут показаны вся его история записей в салон.

Рисунок 2.7 – Страница история

На этом проектирование приложения закончено и можно приступать к реализации непосредственно самому написанию кода.

2.4. Программно-технические средства, необходимые для разработки Веб-приложения

При разработке программного средства необходимо определиться с компонентами и технологиями, которые необходимы для реализации дипломного проекта.

Архитектура данного проекта – это веб-приложение. Архитектура приложения во многом предопределило и технологии, используемые для построения проекта.

Основные технологии, применяемые при создании приложения:

– серверный язык *JAVA*;

– язык запросов *SQL* (*MySQL*);

– фреймворк *Spring*;

– язык разметки *HTML*;

– таблица стилей *CSS*;

– клиентский язык *JavaScript*.

Каждая из выбранных технологий отвечает за разные аспекты работы программы.

Язык Java появился в 1995 году – 90-е годы были вообще урожайными на новые языки и концепции программирования. В таком Эдеме языков важно было не заблудиться, по

ошибке приняв за Священный Грааль технологию, которая не пройдет испытания временем. Java прошел испытания, хотя и очень долгие. Очень не рекомендуется путать этот язык с JavaScript – они по виду похожи, но это совсем разные языки.

Вероятно, в Java впервые реализовали концепцию того, что язык должен быть максимально изолирован от платформы разработки, чтобы применять его без изменений везде: в компьютерах, часах, сотовых телефонах, бытовой технике. С «железной частью» должна была справляться виртуальная машина (JVM), которая, собственно, и создавалась индивидуально под каждое устройство. Сам же язык был неизменен и в качестве результата выдавал байт-код. С самого начала было известно, что код не может исполняться очень быстро, но многие устройства не требовали высокой скорости исполнения. Кроме того, со временем появились оптимизирующие компиляторы, так что, в среднем, программа на Java работает раза в 2-3 медленнее, чем на C++. Постоянное сравнение с C/C++ здесь не случайно: многие современные языки взяли за основу его конструкции и синтаксис, так что, бывает, узнать сходу язык очень трудно. Вместе с тем, Java с тех пор сильно «размножилась», и даже J#, J и прочие аналоги являются не родными братьями, а лишь подобием.

Сама идея языка, вполне, кстати, достаточного для создания софта любой сложности, была сначала не понята: был ли, мол, смысл создавать между аппаратурой и кодом промежуточные слои исполняющих машин? Со временем сомнения рассеялись: появилась мультиязычная платфома .

NET, и даже в Windows появились слои – аппаратно-зависимые, платформо-независимые. Самое же простое объяснение – софт стал очень сложным, а программисты очень ленивыми, чтобы переписывать программы под каждый отдельный аппарат.

Но вернемся к языку. Как уже говорилось, чем-то он похож на C++, чем-то на старый добрый Бейсик. Нет сейчас ни одного языка, который бы не хвалился своими возможностями ООП, и Java здесь не отличается от канонов: классы и объекты здесь используются везде, даже в самых примитивных задачах вроде вывода строки на экран. Из особенностей можно отметить, что все объекты в языке создаются только динамически, а все функции являются методами классов. Множественное наследование не поддерживается, как в C++, как и «опасные» указатели. ООП дает много преимуществ, но и требует слишком многого – в случае Java памяти устройства никогда не будет слишком много. В остальном же, имеются библиотеки классов для практически всех задач; преимущественно – под написание клиентских и серверных приложений. Хозяин Java – Oracle – успешно использует язык для использования в разработках своей одноименной СУБД. На сегодняшний день язык считается наиболее востребованным на рынке.

Spring Framework (или коротко Spring) — универсальный фреймворк с открытым исходным кодом для Java-платформы. Также существует форк для платформы .NET Framework, названный Spring.NET.

Первая версия была написана Родом Джонсоном, который впервые опубликовал её вместе с изданием своей книги «Expert One-on-One Java EE Design and Development» (Wrox Press, октябрь 2002 года).

Фреймворк был впервые выпущен под лицензией Apache 2.0 license в июне 2003 года. Первая стабильная версия 1.0 была выпущена в марте 2004. Spring 2.0 был выпущен в октябре 2006, Spring 2.5 — в ноябре 2007, Spring 3.0 в декабре 2009, и Spring 3.1 в декабре 2011. Текущая версия — 5.2.x.

Несмотря на то, что Spring не обеспечивал какую-либо конкретную модель программирования, он стал широко распространённым в Java-сообществе главным образом как альтернатива и замена модели Enterprise JavaBeans. Spring предоставляет бóльшую свободу Java-разработчикам в проектировании; кроме того, он предоставляет хорошо документированные и лёгкие в использовании средства решения проблем, возникающих при создании приложений корпоративного масштаба.

Между тем, особенности ядра Spring применимы в любом Java-приложении, и существует множество расширений и усовершенствований для построения веб-приложений на Java Enterprise платформе. По этим причинам Spring приобрёл большую популярность и признаётся разработчиками как стратегически важный фреймворк.

Следующие две технологии – это *HTML* и *CSS*, которые предопределены архитектурой проекта. Имеется единственное обоснование их выбора – это веб-архитектура разрабатываемого программного средства.

HTML – язык гипертекстовой разметки, который используется для структурирования и отображения веб-страницы и ее контента. Для разработки данного программного средства будет применяться версия этого языка *HTML*5.

CSS – это формальный язык описания внешнего вида документа, написанного с использованием языка разметки. Преимущественно используется как средство описания, оформления внешнего вида веб-страниц, написанных с помощью языков разметки *HTML*. Применяемая версия данной технологии – *CSS*3.

Последняя из основных применяемый технологий – это *JavaScript*. С помощью данного языка программирования, можно создавать динамически обновляемый контент, управлять мультимедиа, анимировать изображения. Что касается обоснований выбора в качестве клиентского языка *JavaScript*, то здесь выбор достаточно очевиден, ведь конкурентов у данного языка в данной сфере практически нет. Лишь некоторые достоинства данного языка:

возможность навигации и управления по *DOM HTML*-страницы, возможность управления браузером, гибкость подхода объектно-ориентированного программирования, поддержка асинхронности.

Для проектирования программного средства используется язык графического описания *UML*.

UML – язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования бизнес-процессов, системного проектирования и отображения организационных структур.

UML является языком широкого профиля, это – открытый стандарт, использующий графические обозначения для создания абстрактной модели системы, называемой *UML*-моделью. *UML* был создан для определения, визуализации, проектирования и документирования, в основном, программных систем. *UML* не является языком программирования, на основании *UML*-моделей возможна генерация кода [16].

Следующим этапом данного пункта будет рассмотрение структуры программы посредствам следующих *UML*-диаграмм:

– диаграмма компонентов;

– диаграмма развёртывания;

– диаграмма последовательности.

Диаграмма компонентов – это элемент языка моделирования *UML*, статическая структурная диаграмма, которая показывает разбиение программной системы на структурные компоненты и связи (зависимости) между компонентами.

В качестве физических компонентов могут выступать файлы, библиотеки, модули, исполняемые файлы, пакеты и другое.

Диаграмма компонентов для текущего проекта представлена на рисунке 4.1. Как видно, система состоит из трёх компонентов:

– СУБД *MySQL*;

– серверное приложение;

– браузер клиента.

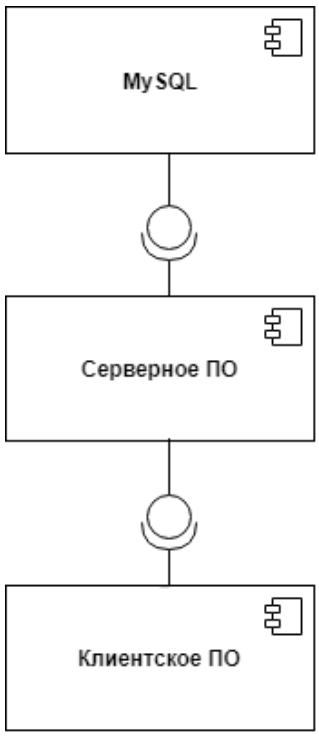

Рисунок 2.8 – Диаграмма компонентов

Каждый аспект отвечает за свой функционал. СУБД *MySQL* – за хранение и выборку данных, серверное приложение – за логику системы, а браузер клиента – за отображение информации.

Для выполнения дипломного проекта использовался персональный ЭВМ, имеющий следующие характеристики:

Процессор: Intel® Core7

Память: 16 ГБ

Видеокарта: NVIDIA GeForce 9800 GT

Жест.диски: ST3640323AS ATA Device 640Gb

Опер.система: Microsoft Windows 10 Профессиональная (64 разрядная операционная система).

2.5. Описание ограничений доступа к данным

Доступ к данным определяется по типу роли пользователя в системе. Работу с ролями берет на себя Spring Security, нас остается только правильно настроить конфигурационный файл.

Spring Security – это платформа для приложений Java и Java EE, которая предоставляет специализированные механизмы для построения систем авторизации и аутентификации. Она представляет собой настраиваемую систему для проверки подлинности и контроля доступа к

определенным ресурсам приложений Java и Java EE, а также защиты корпоративных приложений, созданных с помощью Spring Framework. Впервые разработку данной платформы начал Бен Алекс (Ben Alex) в 2003 году, называлась она «Acegi Security», после этого в 2004 году был выпущен ее первый релиз.

Однако долго самостоятельным проектом она не просуществовала и была поглощена компанией, как следствие этого она стала его официальным дочерним проектом. Под юрисдикцией компании «Spring» платформа была публично представлена под новым именем «Spring Security» версии релиза 2.0.0 в апреле 2008 года.

2.6 Описание используемых библиотек и элементов управления

В качестве библиотек использовались следующие технологии:

JQuery HTML — это библиотека, предназначенная чтобы «писать меньше, а делать больше«. Это не язык программирования, а инструмент, используемый для того, чтобы упростить реализацию общих задач JavaScript.

Owl Carousel - jQuery плагин с поддержкой touch, позволяющий создать отзывчивый (адаптивный) слайдер (карусель). Это очень удобны, простой и адаптивный плагин для создания слайдеров, каруселей и т.д.

TinyMCE — платформонезависимый Javascript HTML визуальный веб-редактор, выпускается как Open Source. TinyMCE очень легко интегрируется в различные CMS.

Редактор позволяет очень многое. С его помощью можно вставлять рисунки, таблицы, применять стили оформления текста. Можно выполнять операции, присутствующие в более мощных редакторах. Вдобавок к этому существует множество плагинов.

2.7. Описание используемых функций и процедур

Так как мы используем шаблон MVC, идет отдельно разработка классов модели (сущности), отдельно вида (страниц для браузера) и контроллеров (обработки действий пользователя).

Так как данные в шаблоне будут изменяться по полученным из бд, нам нужно создать класс-сущность для хранения этих данных.

Сущность (entity) – это реальный или представляемый тип объекта, информация о котором должна сохраняться и быть доступна. Поля сущности совпадают с полями таблицы, в которой эти данные хранятся в базе данных.

Пример класса сущности отображен в листинге ниже.

Листинг 1. Сущность пользователя

```
@Entity
@Table(name = "myUsers")
@Data
public class User extends AbstractEntity {
   private String fio;
   private Date dateBorn;
   private String email;
   private String phone;
   @Column(unique = true)
   private String username;
   private String password;
   @OneToMany(mappedBy = "user", fetch = FetchType.LAZY)
   private List<Order> orders = new ArrayList<Order>();
   @ManyToOne
   @JoinColumn(name = "role_id")
   private Role role;
   public User() {
      super();
   }
   public List<String> getRoleListNames() {
      List<String> roleNames = new ArrayList<String>();
      roleNames.add(role.getName());
      return roleNames;
   }
   @Override
   public String toString() {
      return "User [fio=" + fio + ", dateBorn=" + dateBorn + ", email=" + email + ", phone="
+ phone + ", username="
            + username + "]";
   }
}
```

По подобию кода выше пишутся сущности для всех остальных объектов, а именно: роли, категории, услуги, заказа.

После написания сущностей идет описание сервисов – классов, которые взаимодействуют с базой данных, читая из нее данные и отдавая нам, ила наоборот – записывая в нее.

Код классов для сервисов также похож между собой, так что покажем только абстрактную реализацию сервиса.

Листинг 2. Абстрактная реализация сервиса.

```
public class CrudImpl<Entity> implements CrudService<Entity> {
public JpaRepository<Entity, Long> repository;
public boolean create(Entity entity) {
        try {
                repository.saveAndFlush(entity);
        } catch (Exception e) {
                e.printStackTrace();
                return false;
        }
        return true;
}
public Entity read(long id) {
        return repository.findById(id).isPresent() ? repository.findById(id).get() : null;
}
public boolean update(Entity entity) {
        try {
                repository.saveAndFlush(entity);
        } catch (Exception e) {
                e.printStackTrace();
                return false;
        }
        return true;
}
public boolean delete(long id) throws Exception {
        try {
                repository.deleteById(id);
        } catch (Exception e) {
                e.printStackTrace();
                return false;
```

```
        }
        return true;
    }
    public CrudImpl(JpaRepository<Entity, Long> repository) {
        super();
        this.repository = repository;
    }
    public List<Entity> getAll() {
        List<Entity> result = repository.findAll();
        if (result.size() > 0) {
            return result;
        } else {
            return new ArrayList<Entity>();
        }       }}
```

После реализации сервисов, остается только 2 шага до окончания разработки приложения – это написание контроллеров и представлений. Контроллер – это специализированный класс, который отвечает на запросы пользователей по определенным ссылкам. Обычно они именуются так: СтраницаController, где страница – это имя страница, за которую отвечает контроллер. В контроллерах опишем ответы на запросы пользователя по взаимодействию с базой данных, т.е. ответы на запросы типа /read, /create/ edit. Ниже показан контроллер пользователя в качестве примера. Остальные контроллеры пишутся по его подобию.

Листинг 3. Контроллер пользователя

```
@Controller
@RequestMapping("/user")
public class UserController {
   @Autowired
   private UserServiceImpl service;
   @Autowired
   private PasswordEncoder passwordEncoder;
   @GetMapping
   String get(Model model) {
      List<User> list = service.repository.findAll();
      model.addAttribute("list", list);
      return "user-list";
```

```
}
@RequestMapping(path = {"/edit", "/edit/{id}"})
public String edit(Model model, @PathVariable(name = "id", required =
        false) Long id) {
    if (id != null) {
        User entity = service.read(id);
        model.addAttribute("entity", entity);
    } else {
        model.addAttribute("entity", new User());
    }
    return "user-add-edit";
}
@RequestMapping(path = "/create", method = RequestMethod.POST)
public String createOrUpdate(User entity) throws Exception {
    if (entity.getId() == null)
        service.userRegistration(entity);
    else {
        User read = service.read(entity.getId());
        read.setDateBorn(entity.getDateBorn());
        read.setEmail(entity.getEmail());
        read.setFio(entity.getFio());
        read.setPhone(entity.getPhone());
        read.setPassword(passwordEncoder.encode(entity.getPassword()));
        service.update(read);
    }
    return "redirect:/user";
}
@RequestMapping(path = "/delete/{id}")
public String delete(Model model, @PathVariable("id") Long id) throws Exception {
    service.delete(id);
    return "redirect:/user";
}
@RequestMapping(path = "/history")
public String history(Model model, Principal principal) {
    User user = service.findByUsername(principal.getName());
```

```
            model.addAttribute("list", user.getOrders());
            return "history";
        }
    }
```

Остался последний шаг – связывание шаблонов и контроллеров. Для этого воспользуемся шаблонизатором Thymeleaf, который работает в связке с Spring.

Суть его проста – те данные, что мы передавали в модель отображения, таймлиф можем выводить простой командой th:text=”${}”. В скобках пишется имя переменной. Обращение идет как к переменным класса. В нужных местах вставляем этот код. На нашем лендинге.

Так, для отображения шапки других шаблонов будут использоваться строки листинга 4.

Листинг 4. Изменение текста страницы

```
<table class="table table-striped table-responsive-md">
<thead>
        <tr>
                <th>ФИО</th>
                <th>Дата рождения</th>
                <th>Почта</th>
                <th>Телефон</th>
                <th>Логин</th>
        </tr>
</thead>
<tbody>
        <tr th:each="entity : ${list}">
                <td th:text="${entity.fio}"></td>
                <td th:text="${entity.dateBorn}"></td>
                <td th:text="${entity.phone}"></td>
                <td th:text="${entity.email}"></td>
                <td th:text="${entity.username}"></td>
                 <td class="btnInTable widthLastCol" >
                        <a th:href="@{/user/edit/{id}(id=${entity.id})}" class="btn btn-primary ">
                                <i class="fas fa-edit">Редактировать</i>
                        </a>
                </td>
```

```
                    <td class="btnInTable widthLastCol">
                        <a th:href="@{/user/delete/{id}(id=${entity.id})}" class="btn btn-primary">
                            <i class="fas fa-trash">Удалить</i>
                        </a>
                    </td>
                </tr>
            </tbody>
            </table>
```

2.8 Организация базы данных

Spring Framework позволяет генерировать базу данных на основе правильно описанных сущностей и взаимодействий между ними. Иначе говоря, после создания классов приложения, фреймворк при запуске приложения сам создаст базу данных, если она еще не была создана. Таблицами такой базы будут классы сущности, названиями таблиц будут названия сущностей. Тип и размер полей фреймворк также сам задаст на основе типов класса, если они не помечены аннотацией Length (задание размера вручную).

Схема базы данных отображена на рисунке ниже.

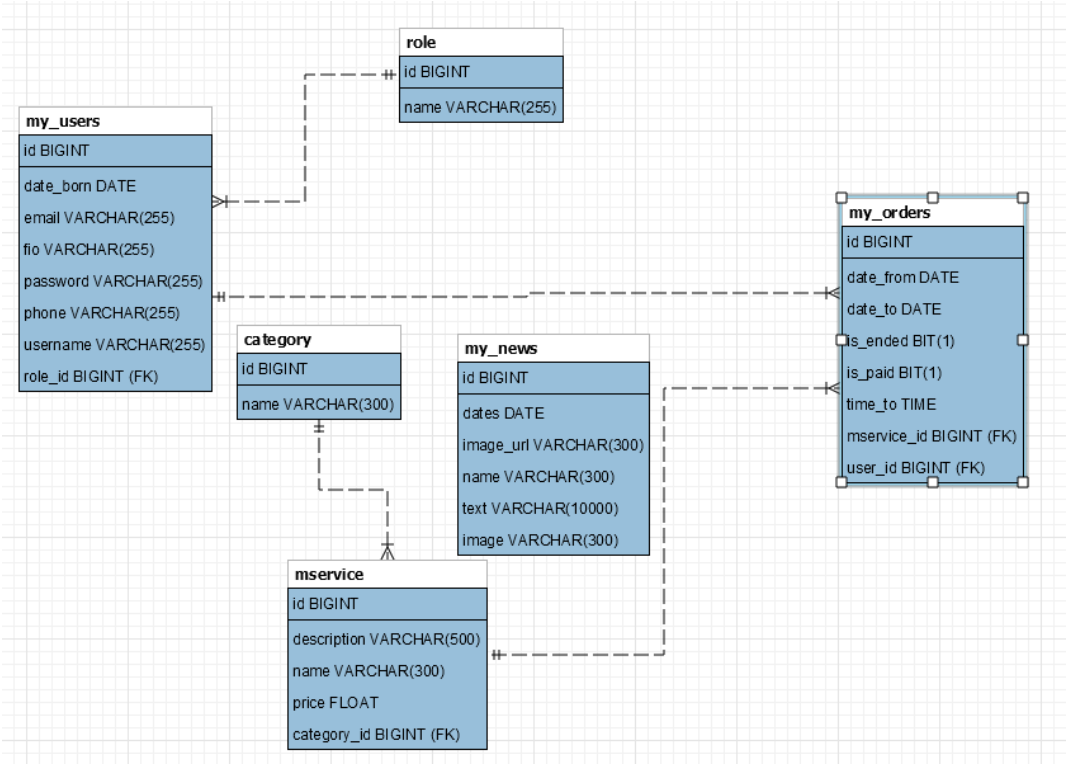

Рисунок 2.9 – Схема базы данных

Данные и путь для доступа к базе данных задаются в файле настройки application.properties.

Листинг 5 – Свойства доступа к базе данных

```
spring.jpa.properties.hibernate.enable_lazy_load_no_trans=true
spring.datasource.dbcp2.timeBetweenEvictionRunsMillis = 3000
spring.datasource.dbcp2.minEvictableIdleTimeMillis = 6000
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://${MYSQL_HOST:localhost}:3306/chik_chik_barbershop?createDatabaseIfNotExist=true&serverTimezone=UTC&characterEncoding=utf8
spring.datasource.username=root
spring.datasource.password=1234
spring.datasource.initialization-mode=always
spring.devtools.livereload.enabled=true
```

3 МЕТОДИКА ИСПЫТАНИЙ

3.1 Технические требования

Основные требования:

1. Удобная и интуитивно понятная панель управления сайта.
2. Визуальный редактор для удобного редактирования контента страниц.
3. Исходный код системы управления должен быть открытым и хорошо читаемым, чтобы при необходимости была возможность более глубокой оптимизации.

Адреса страниц

1. Каждая страница должна иметь свой уникальный канонический адрес
2. Внутренние ссылки должны быть абсолютными.

Ссылки

1. Для наиболее важных страниц количество исходящих ссылок должно стремиться к минимуму, а входящих к максимуму.
2. Общее количество ссылок на странице не должно превышать 250 — 300 единиц.
3. Техническая ссылочная масса должна быть сведена к минимуму или реализована без использования ссылок в разметке.

3.2 Функциональное тестирование

Для проверки работы спроектированной автоматизированной системы, разработаны тесты, с помощью которых можно оценить корректную работу веб-сервиса. В таблицах 3.1-3.2 приведены результаты тестирования.

Таблица 1.1 – Набор тестов для автоматизированной системы

Место проведения теста	Содержание теста	Ожидаемый результат	Отметка о прохождении теста
Главная страница	Загрузка главной страницы	Вывод меню в зависимости от роли и списка товаров	Да
Страница добавления категории услуги	Перейти на страницу добавления. Ввести название. Нажать кнопку отправить	В списке категорий услуг отображается только что созданная категория	Да
Страница услуги	Перейти на страницу добавления услуги. Заполнить поля, из списка категории выбрать добавленную	В списке товаров отображается только что созданная услуга с новой категорией.	Да

	ранее категорию. Нажать кнопку отправить.		
Страница заказов	Перейти на страницу оформления заказа, заполнить поля и выбрать добавленную ранее услугу. Оформить заказ.	На странице истории заказов отобразится текущий заказ и статус его оплаты и выполнения.	Да

Таблица 4.2 – Набор тестов отображения информации

Место проведения теста	Содержание теста	Ожидаемый результат	Отметка о прохождении теста
Все страницы	Перейти на любую из вкладок в произвольном порядке.	Корректное отображение информации, отображение того, чего ожидает пользователь.	Да
Все вкладки	Увеличение масштаба отображения средствами браузера.	Появление скроллов справа и снизу, корректное отображение информации.	Да
Все вкладки	Обновление страницы используя средства браузера.	Состояние веб-сервиса до обновления и после не изменилось.	Да
Меню браузера	Нажатие функциональных кнопок браузера «Назад» и «Вперед» в произвольный момент работы веб-приложения.	Корректное поведение веб-сервиса. Отображение информации.	Да

4 ПРИМЕНЕНИЕ

4.1 Назначение веб-приложения

Главное назначение создания сайта-визитки — первичное ознакомление потенциальных клиентов с компанией. Отличительные черты — информативность, лаконичность, индивидуальность, привлекательный дизайн. Сайт не сложен в разработке, содержит не более 10-ти страниц с понятной навигацией: пользователь должен быстро находить нужную информацию. Такой сайт отличает и простота в обслуживании. От владельца требуется только обновлять контакты и иногда загружать новости и прайс-листы, если это предусмотрено разделами.

4.2 Программно-аппаратное обеспечение сервера и клиента

Запуск разработанного Web-приложения и надёжность его работы без сбоев должно обеспечивать аппаратное обеспечение. Любая компьютерная программа для нормальной работы должна располагать определенным количеством ресурсов и, если таких ресурсов по каким-либо причинам не хватает, программа может потерять часть своей функциональности, либо при критической нехватке ресурсов полностью утратить работоспособность.

Управление Web-приложением осуществляется посредством навигационного меню, поэтому для полноценного функционирования программного товара необходимо наличие манипулятора мышь.

Технические характеристики компьютера, на котором проводилось тестирование:

Процессор

Платформа (кодовое название)	Intel Coffee Lake (2018)
Процессор	Intel Core i7
Модель процессора	Intel Core i7 8750H
Количество ядер	6
Тактовая частота	2 200 МГц
Turbo-частота	4 100 МГц
Энергопотребление процессора (TDP)	45 Вт
Встроенная в процессор графика	Intel UHD Graphics 630

Оперативная память

Тип оперативной памяти	DDR4
Частота оперативной памяти	2666 МГц
Объём памяти	16 ГБ

Хранение данных

Конфигурация накопителя	HDD 1000 ГБ + SSD 128 ГБ
Тип накопителя	HDD+SSD
Ёмкость накопителя	1000+128 ГБ
Скорость вращения	5400 RPM
Интерфейс установленного SSD	SATA

Приветствуется изменение указанных характеристик в сторону увеличения. Изменение характеристик в сторону уменьшения не желательно, поскольку это приведет к значительному снижению работоспособности Web-приложения.

4.3 Руководство пользователя

Для запуска проекта необходимо выполнить следующие пункты

1) Установить JDK версии 8 и выше
2) Установить Maven версии 3.0.6 и выше
3) Установить MySQL 8. При установке указать имя пользователя “root” и пароль “1234”
4) Открыть консоль. В консоли перейти в папку проекта
5) В консоли написать команду “mvn eclipse:eclipse”, если при разработке использовалась IDE Eclipse или “mvn idea:idea”, если использовалась IDEA. С помощью этой команды будут загружены все необходимые проекту зависимости, она исполняется лишь раз
6) Для запуска приложения в консоли написать “mvn clean package spring-boot:run”. Эта команда подгрузит оставшиеся зависимости, соберет и запустит проект. Окончанием запуска приложения будет служить соответствующая строка консоли.
7) В любом браузере перейти по адресу http://localhost:8080/

   Ниже отображен процесс работы приложения.

   При переходе на сайт, пользователю будет показана главная страница приложения.

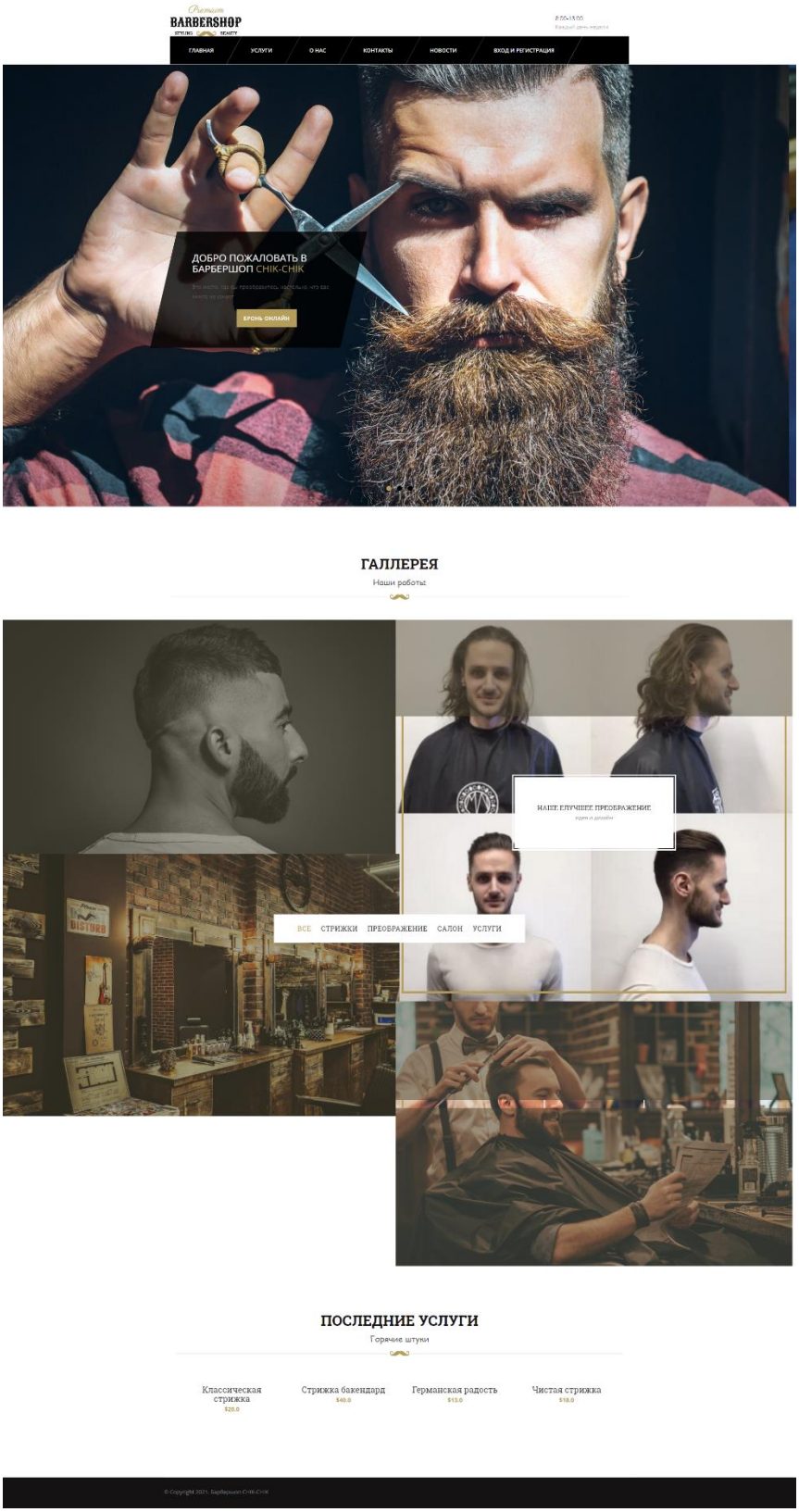

Рисунок 4.1 – Главная страница

С ее помощью можно перейти на любую другую доступную страницу сайта, к примеру страницу услуг.

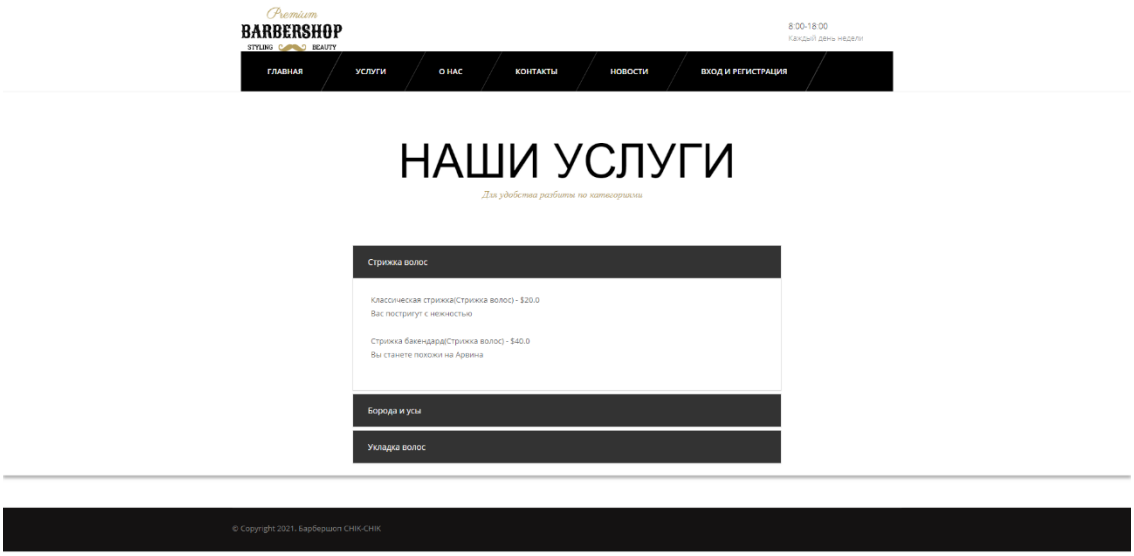

Рисунок 4.2 – Страница услуг

На странице услуг отображаются все услуги барбершопа. Для удобства, они разделены по категориям и оформлены в виде гармошки. Страница о барбершопе также оформлена с использованием гармошки при показе фактов.

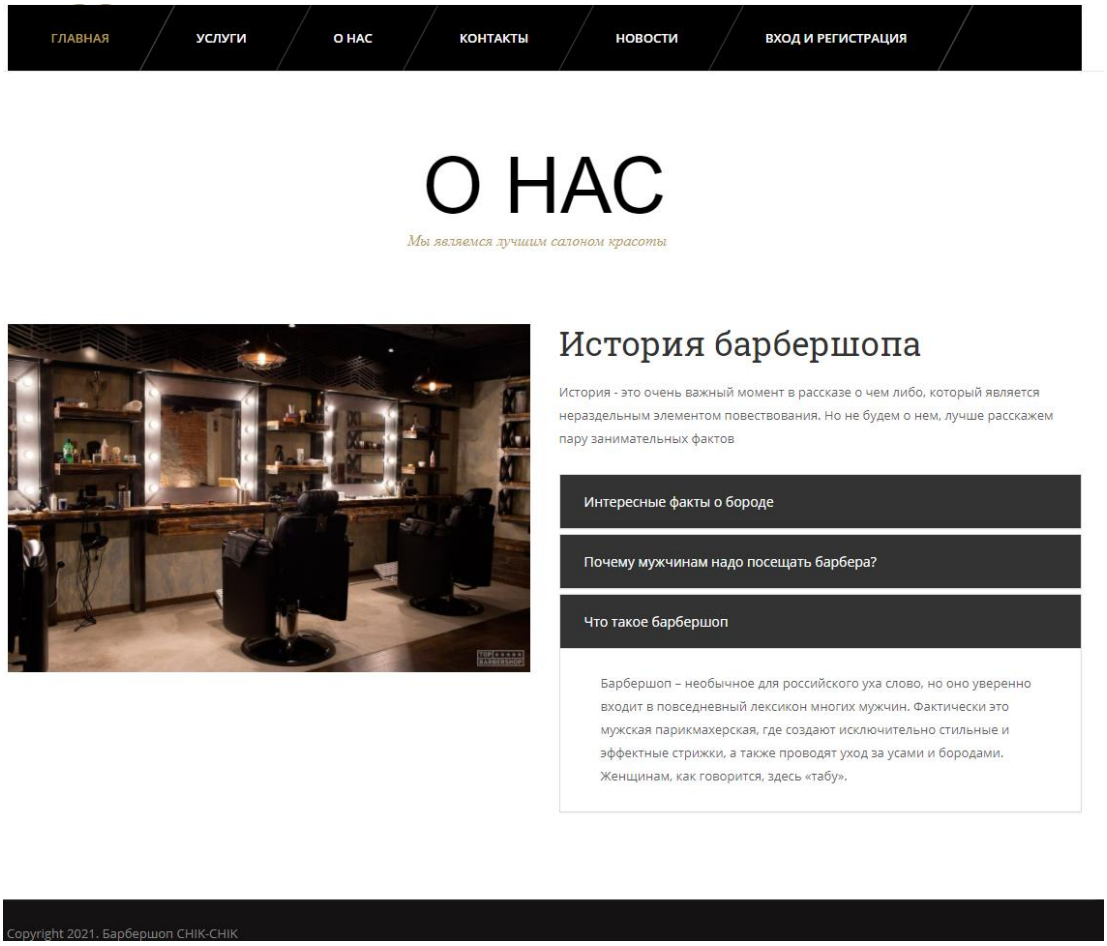

Рисунок 4.3 – Страница о нас

При переходе на страницу контактов, будут отображены все необзодимые для связи данные.

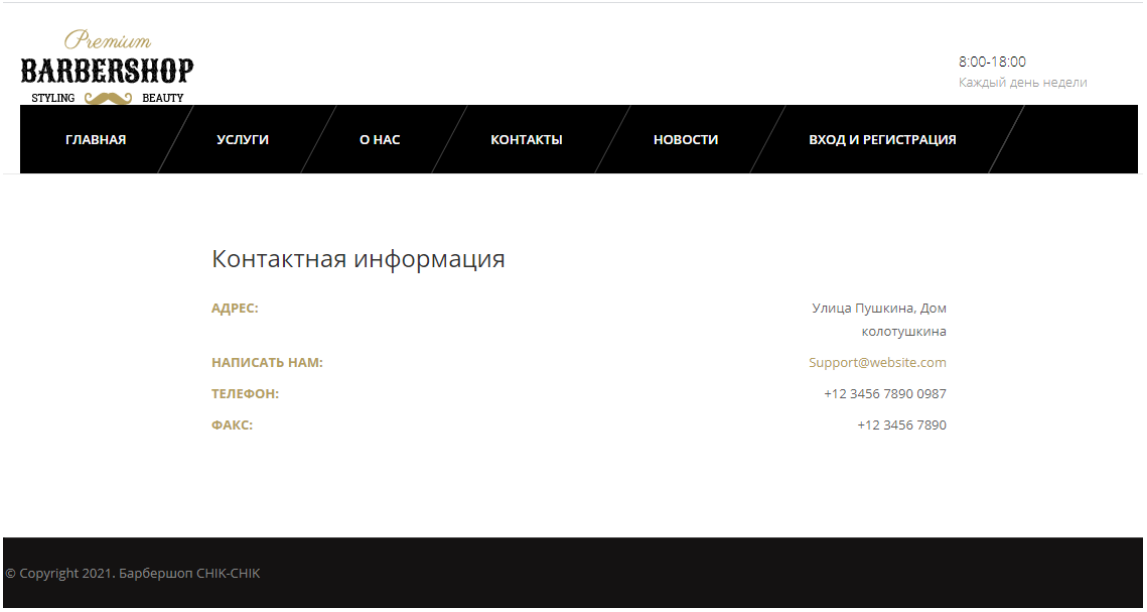

Рисунок 4.4 – Страница контактов

При переходе на страницу новостей, будет отображен список новостей, включающий ее название и дату создания. Для перехода к конкретной новости следует нажать кнопку подробнее или на заглавие новости.

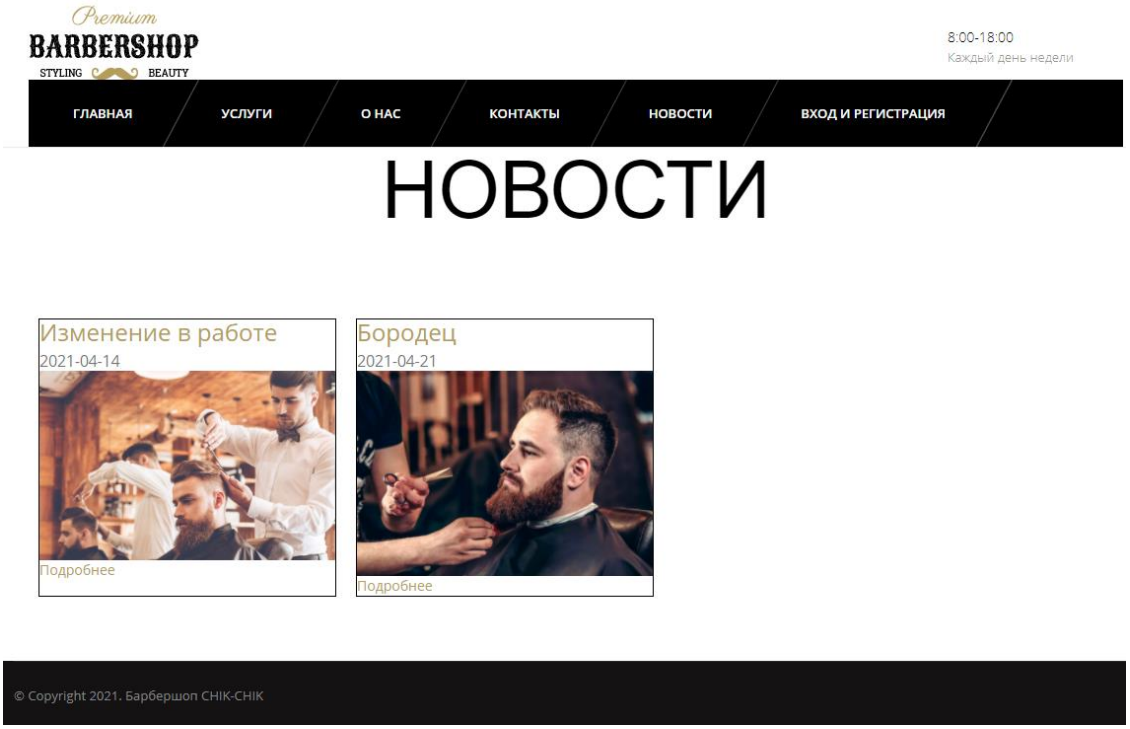

Рисунок 4.5 – Страница новостей

Так новость откроется на новой странице. Текст новости будет отображаться точно с теми же стилями, которые были заданы при ее создании.

Рисунок 4.6 – Подробности новости

На этом функциональность для гостя окончена, для продолжения работы требуется регистрация и вход в систему. Перейдем на страницу регистрации и добавим нового пользователя.

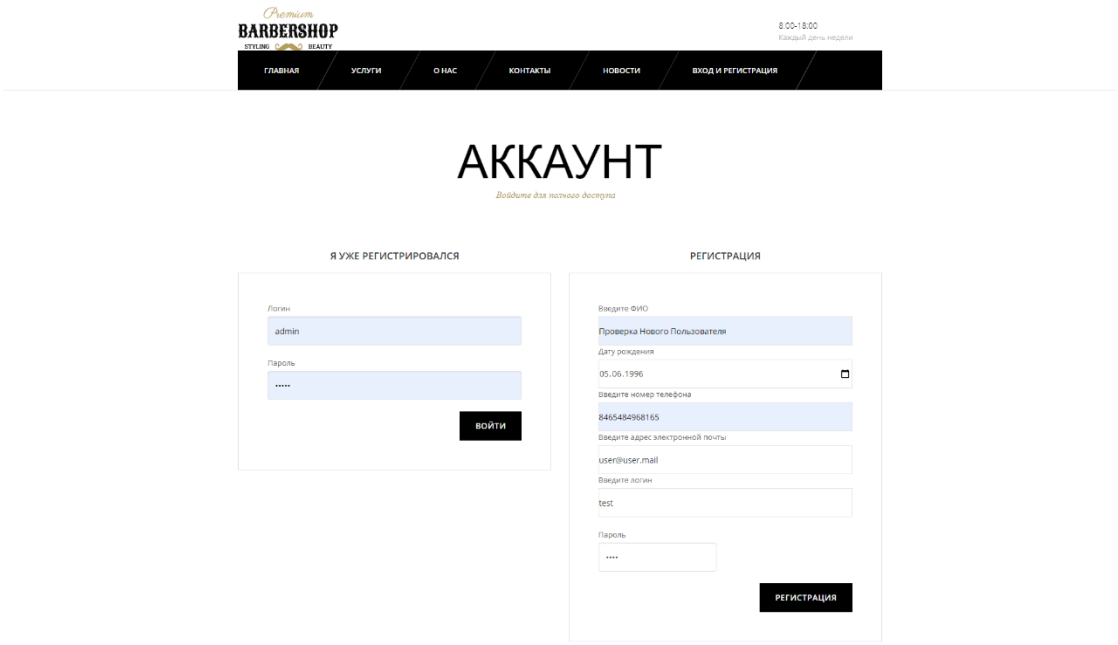

Рисунок 4.7 – Страница входа и регистрации

После заполнения всех данных и отправки их на сервер, мы получим уведомление об успешной регистрации и можем войти в аккаунт

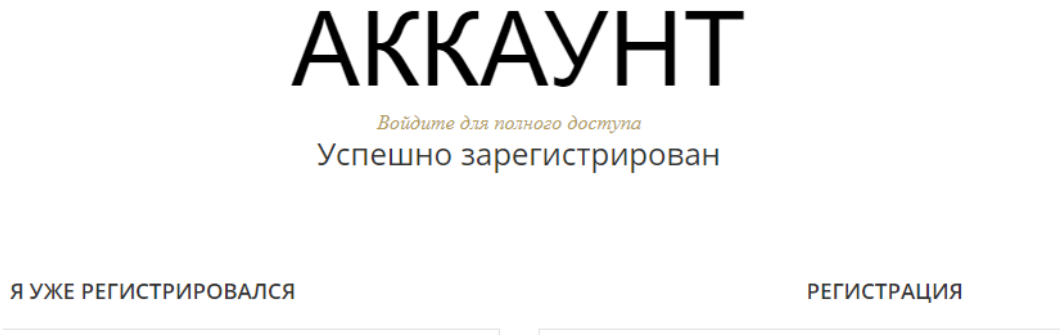

Рисунок 4.8 – Уведомление об успешной регистрации

После входа в аккаунт, изменится меню, отображаемое пользователю, а также добавится возможность бронирования услуги.

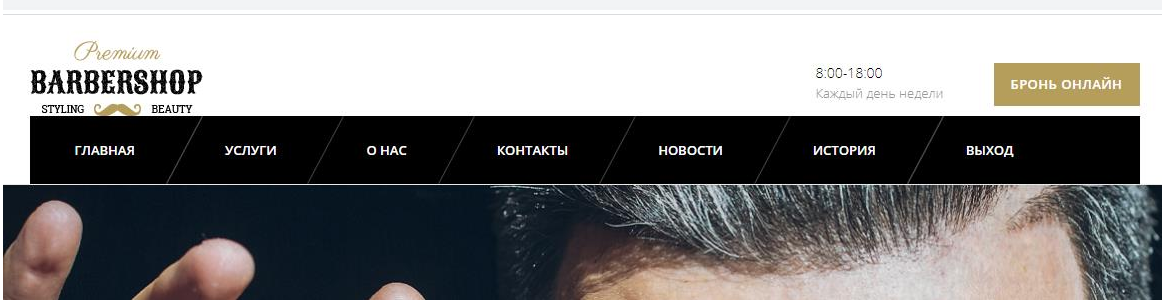

Рисунок 4.9 – Вид меню для пользователя

При переходе на страницу бронирования требуется выбрать на какую дату и время будет бронь и услуги из списка.

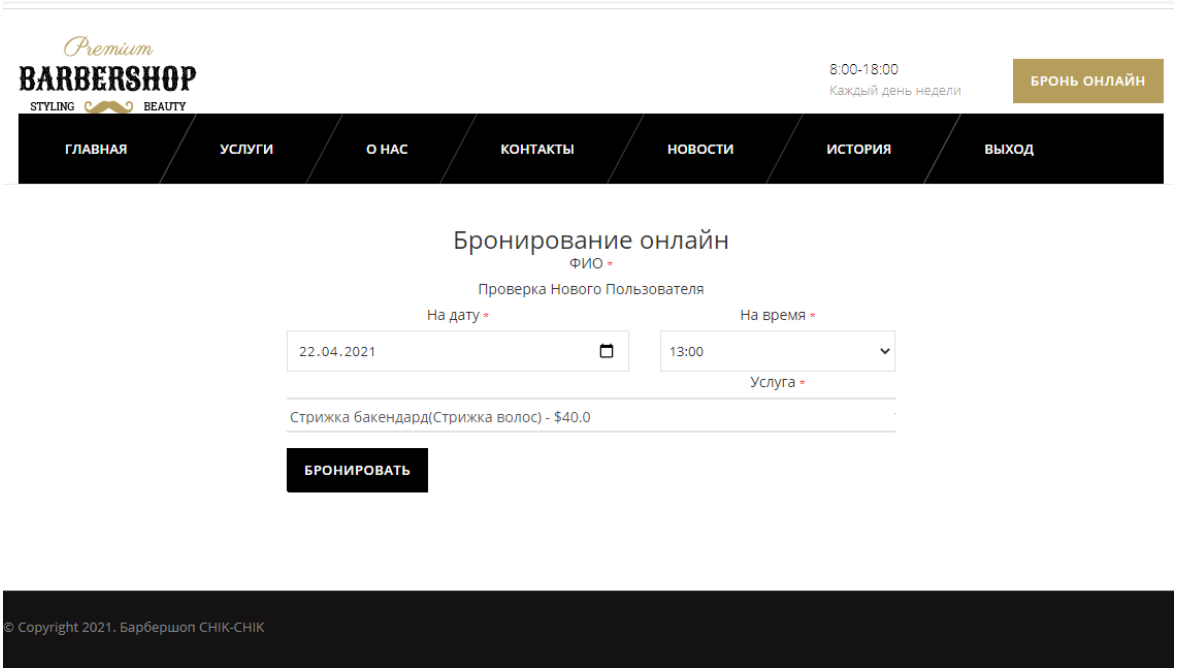

Рисунок 4.10 – Страница бронирования

После бронирования, текущая бронь, а также все предыдущие будут отображены на странице истории, где можно отслеживать их статус.

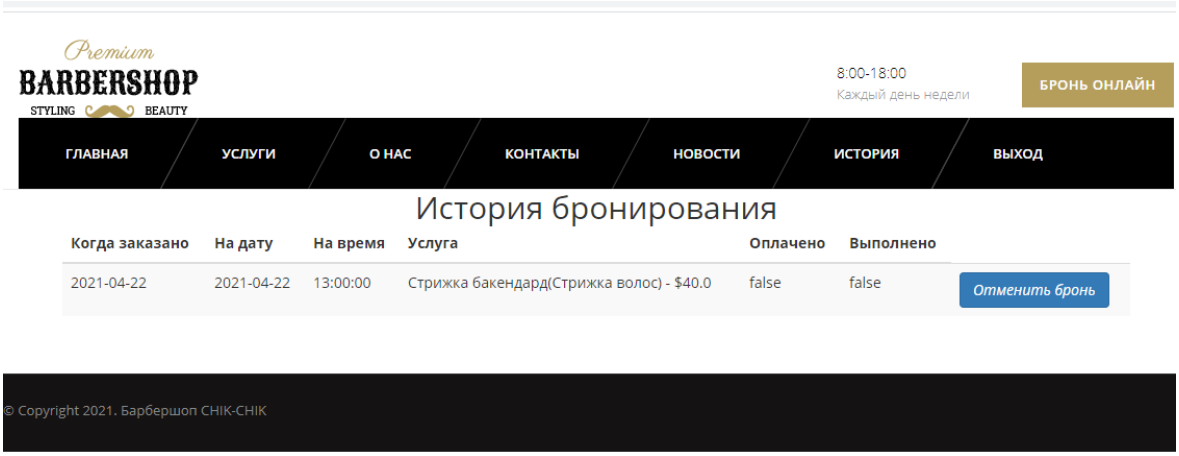

Рисунок 4.11 – Страница истории бронирования

При входе от имени администратора также изменятся пункты меню.

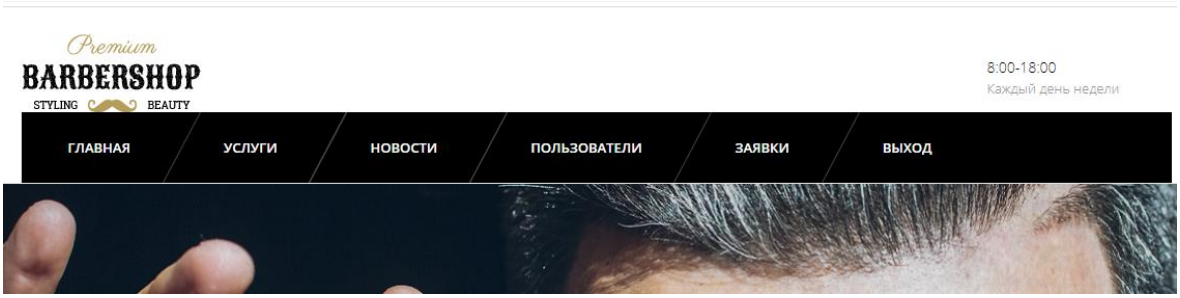

Рисунок 4.12 – Вид меню для администратора

Стоит сказать, что хоть визуально в меню пункт услуги не изменился, изменилось содержание страницы. Так, при переходе на эту страницу будут отображены список категории и список услуг.

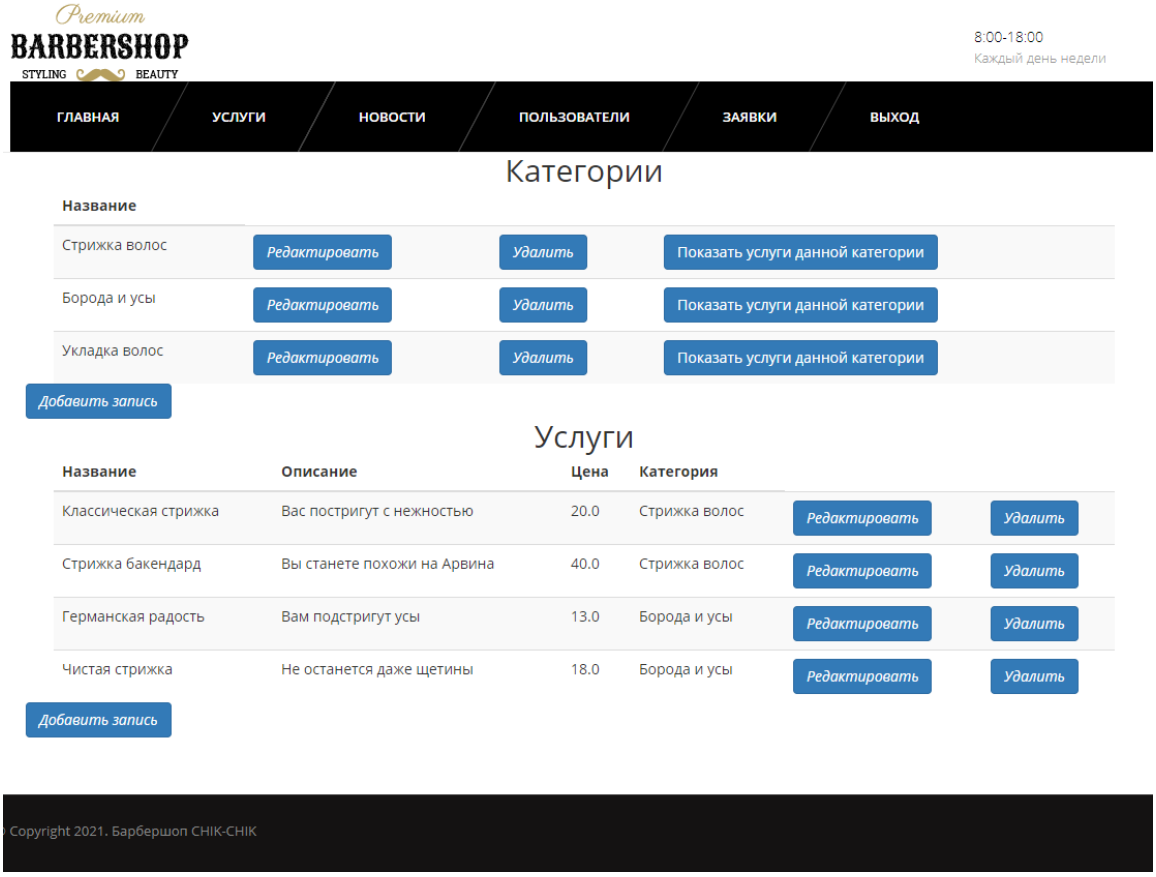

Рисунок 4.13 – Страница услуг для администратора

Для показа услуг только одной категории, нужно выбрать соответствующий пункт меню напротив нужной категории.

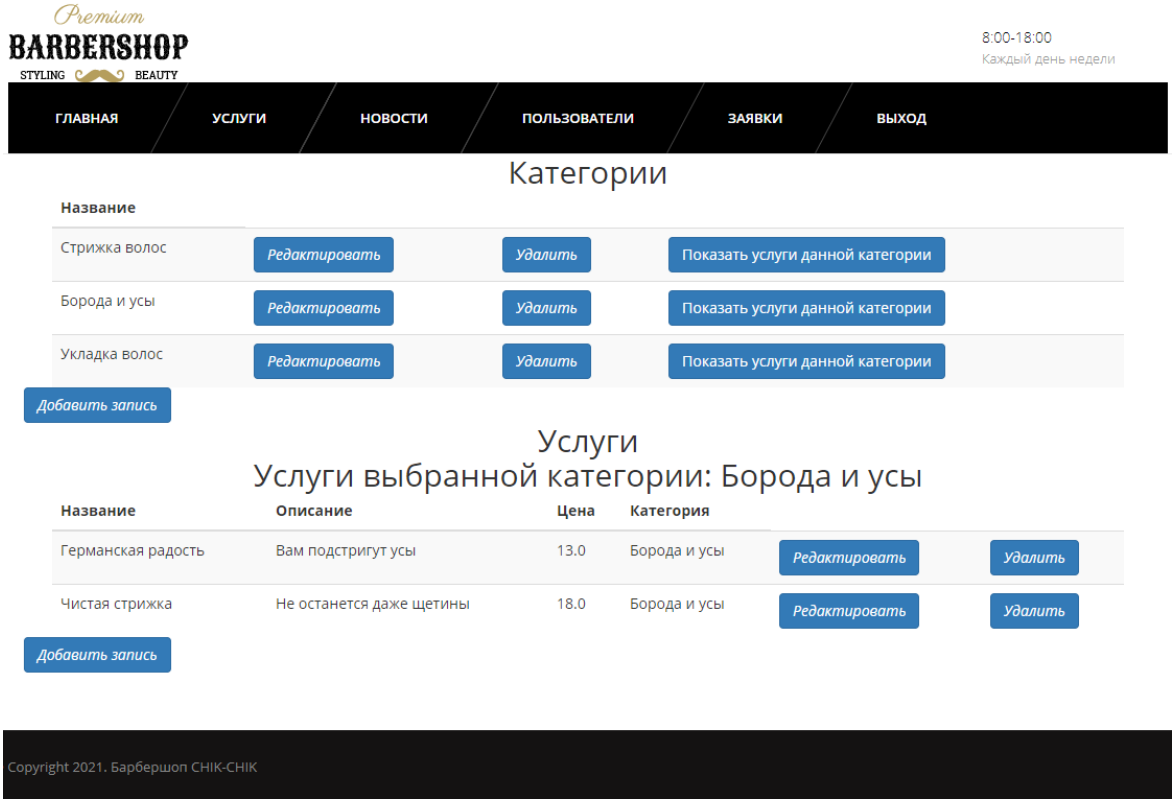

Рисунок 4.14 – Фильтрация услуг выбранной категории

При добавлении категории нужно ввести лишь ее имя.

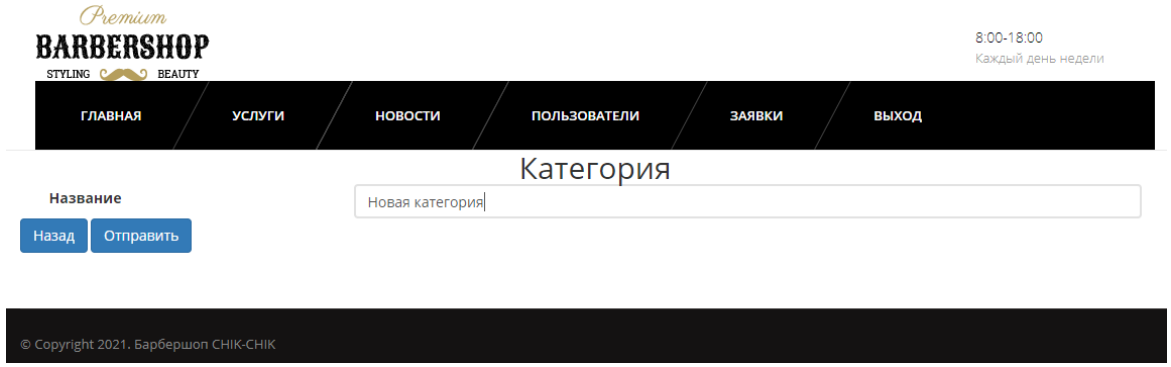

Рисунок 4.15 – Страница добавления категории

Последняя добавленная категория отображается в конце списка.

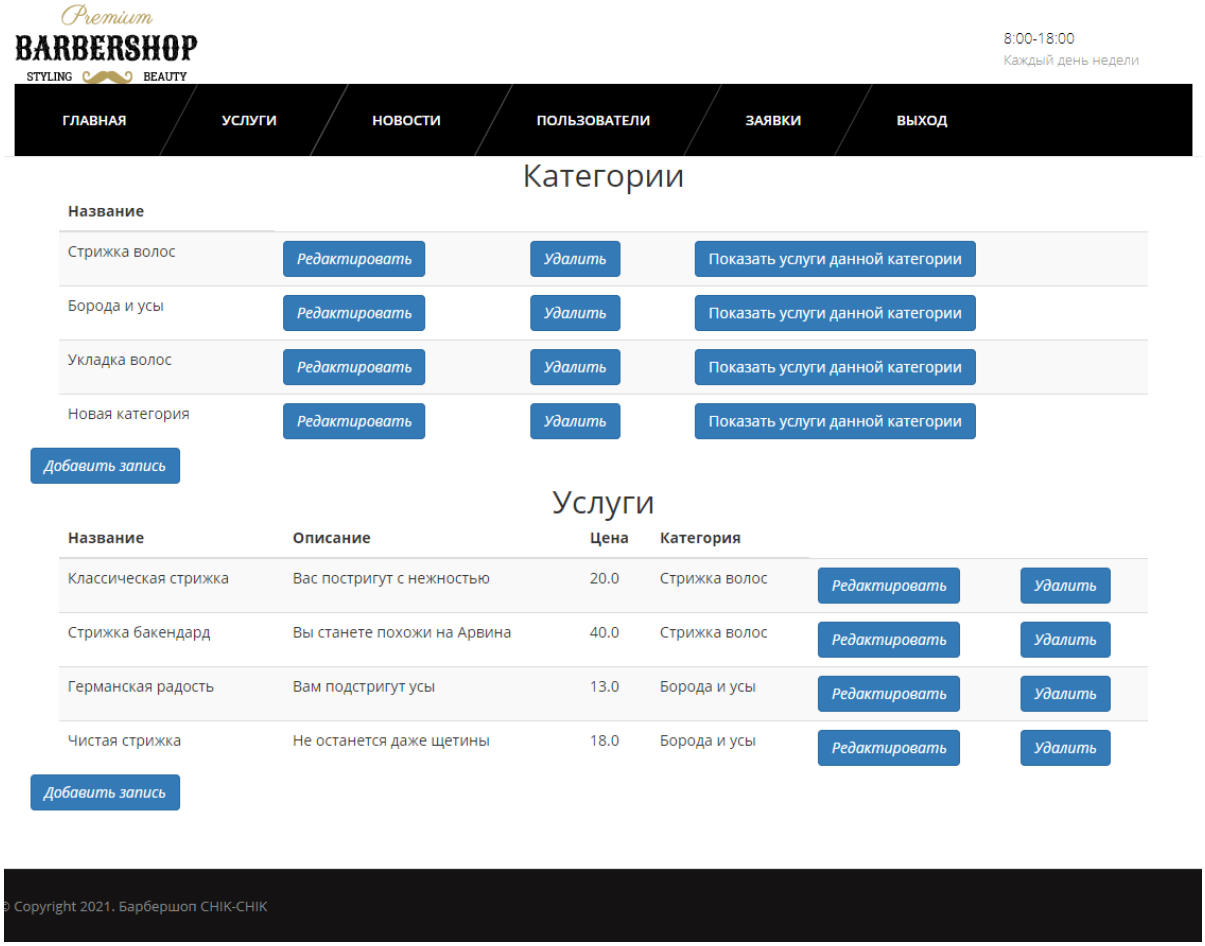

Рисунок 4.16 – Страница услуг после добавления категории

Для добавления услуги стоит нажать кнопку добавления записи под таблицей. Как можно было заметить, недавно добавленная категория также отображается в списке.

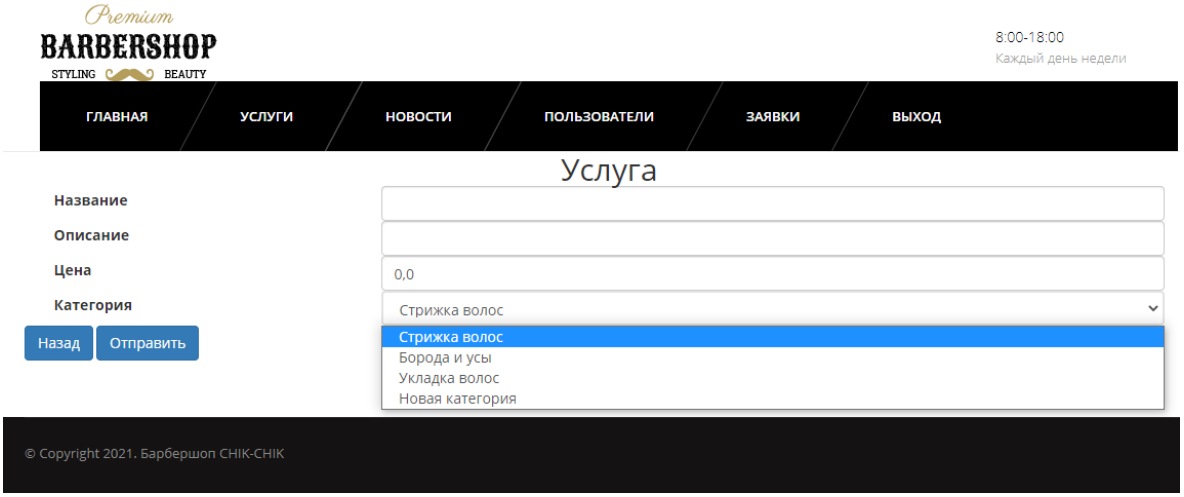

Рисунок 4.17 – Страница добавления услуги

При переходе на страницу новостей от имени администратора, ему доступна работа с записями.

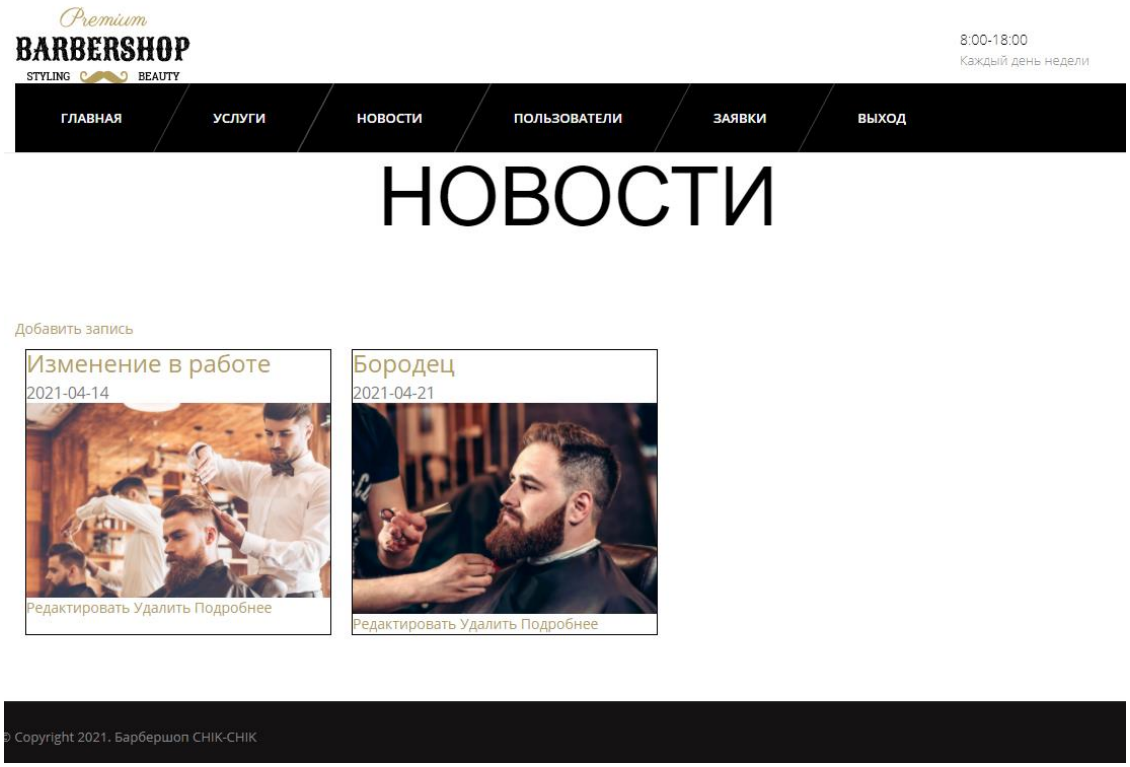

Рисунок 4.18 – Страница новостей для администратора

Стоит сказать, что для добавления текста записи используется специальное поле, которое поддерживает форматирование текста. Т.е. можно даже отформатировать текст в Word, настроить разные цвета, выделения, отступы, все это скопировать в поле и отправить. И текст новости будет отображаться точно так же.

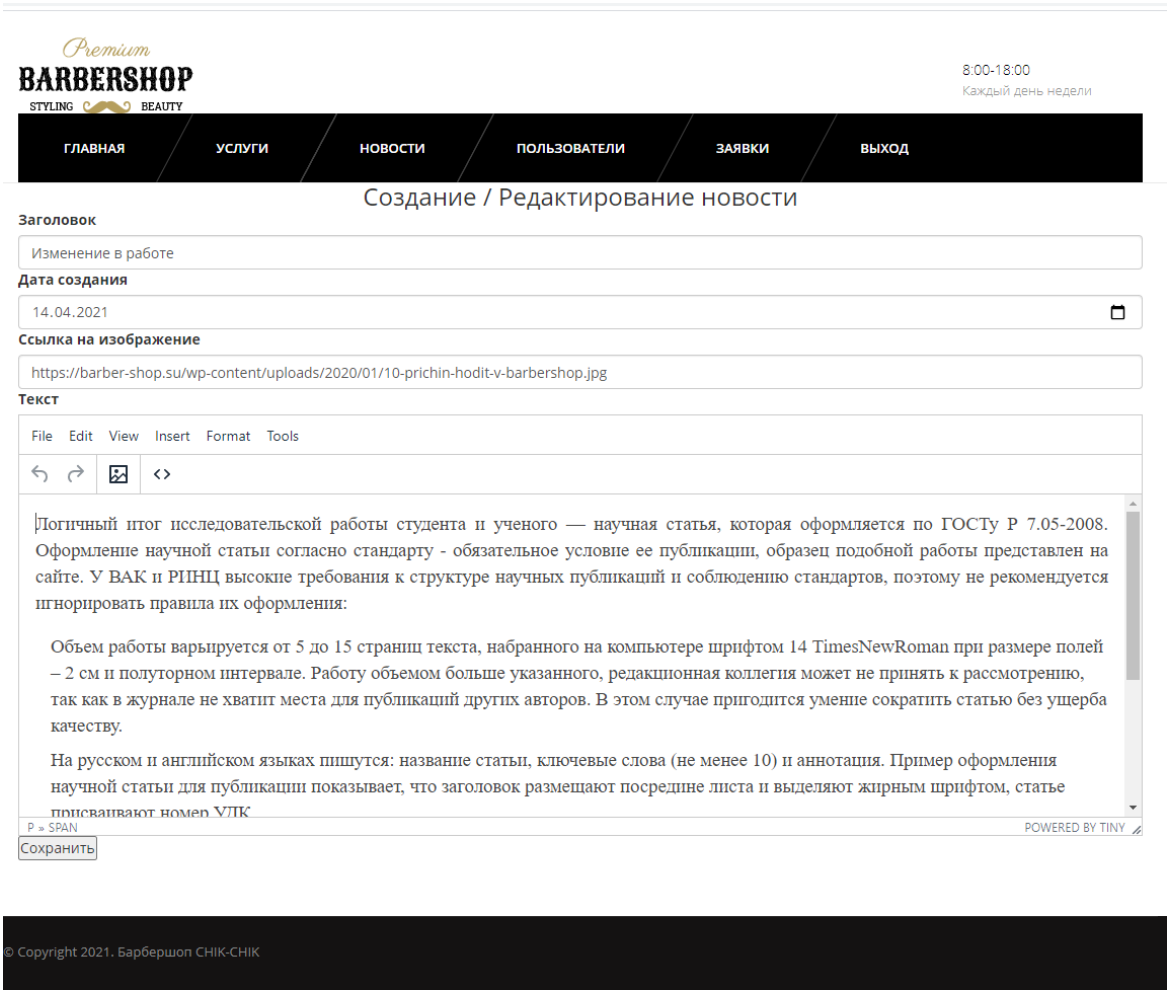

Рисунок 4.19 – Страница редактирования новости

Для работы с пользователями есть отдельная страница, на которой можно просмотреть их список, изменить информацию или же удалить пользователя.

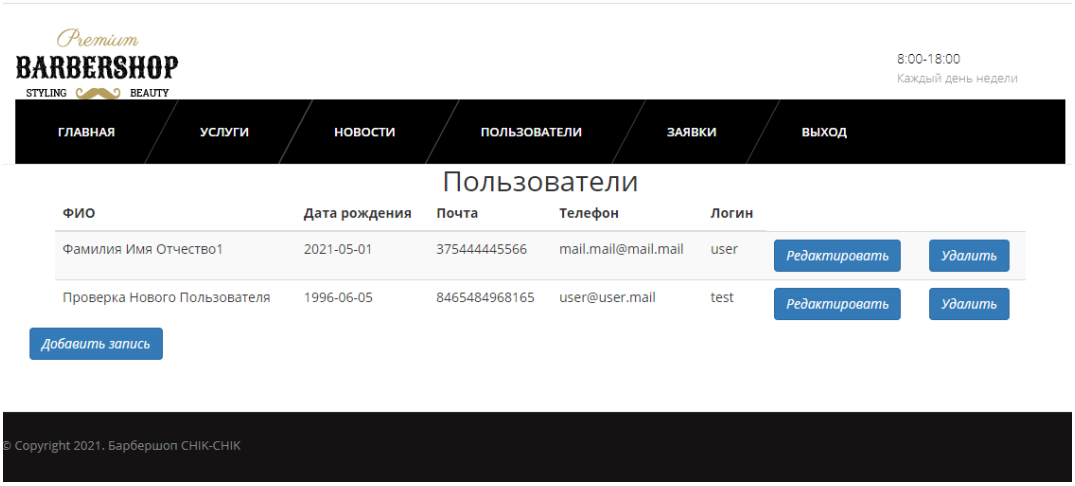

Рисунок 4.20 – Страница управления пользователями

На странице заявок отображены все брони пользователей. Можно как изменить статусы оплаты и выполнения, так и удалить заявку.

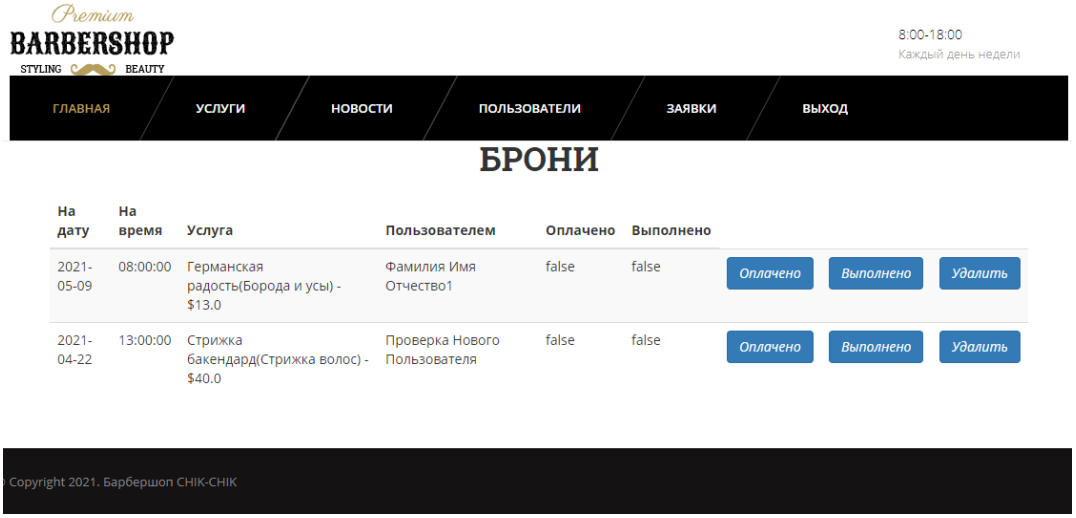

Рисунок 4.21 – Страница заявок

На этом описание руководства пользователя закончено.

5 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ДИПЛОМНОГО ПРОЕКТА

Основная заработная плата исполнителей разработки программного продукта определяется по формуле (5.1)

$$З_0 = \sum_{i=1}^{n} T_{чi} * T_ч * Ф_{Эi} * K, \quad (5.1)$$

где n – количество исполнителей, занятых разработкой ПП;

$T_{чi}$ – часовая тарифная ставка i-го исполнителя, руб.;

$Ф_{Эi}$ – эффективный фонд рабочего времени i-исполнителя, дни;

$T_ч$ – количество часов работы в день, 8 часов;

K – коэффициент премирования (10%).

В настоящий момент базовая ставка в бюджетной организации составляет 185 рублей.

Расчет основной заработной платы предоставлен в таблице 5.1.

Таблица 5.1 – Расчет основной заработной платы

Должность	Месячная тарифная ставка, руб.	Дневная тарифная ставка, руб.	Плановый фонд рабочего времени, дни	Премия, руб.	Заработная плата исполнителей, руб.
Руководитель проекта	720	34,29	50	72	1714,5
Инженер-программист	600	28,57	54	60	1542,78
Итого с учётом премии ($З_0$)					3257,28

Дополнительная заработная плата определяется по формуле (5.2)

$$З_Д = \frac{З_о \cdot Н_Д}{100\%}, \quad (5.2)$$

где $Н_Д$ – норматив дополнительной заработной платы (10%).

Дополнительная заработная плата составит:

$$З_Д = \frac{3257,28 \cdot 10\%}{100\%} = 325,73 \text{ руб.}$$

Отчисления в фонд социальной защиты населения и на обязательное страхование ($З_{сз}$) определяются в соответствии с действующими законодательными актами по формуле (5.3)

$$З_{сз} = \frac{(З_о + З_д) * Н_{сз}}{100\%}, \quad (5.3)$$

где $Н_{СЗ}$ – норматив отчислений в фонд социальной защиты населения и на обязательное страхование (34% + 0,6%).

Отчисления в фонд социальной защиты населения и на обязательное страхование составят:

$$З_{сз} = (3257,28 + 325,73) \cdot \frac{34,6\%}{100\%} = 1239,72 \text{ руб.}$$

Затраты машинного времени на разработку программного средства. Расходы по статье «Машинное время» ($Р_М$) включает оплату машинного времени, необходимого для разработки и отладки ПП, и определяются по формуле (5.4)

$$Р_М = Ц \cdot Т \cdot С_Р, \quad (5.4)$$

где Ц – цена одного машино-часа, руб.;

Т – количество часов в день, 8ч.;

$С_Р$ – длительность проекта, дни.

Стоимость машино-часа в организации составляет 0,45 руб. Разработка проекта занимает 50 дней для руководителя проекта и 54 дней для инженера-программиста.

Затраты по статье «Машинное время» составят:

$$Р_М = 0,45 \cdot 8 \cdot (50 + 54) = 244,40 \text{ руб.}$$

Затраты по статье «Накладные расходы» ($Р_Н$), связанные с необходимостью содержания аппарата управления, определяются по формуле (5.5)

$$Р_Н = \frac{З_о \cdot Н_{РН}}{100\%}, \quad (5.5)$$

где $Н_{РН}$ – норматив накладных расходов (50%).

Затраты по статье «Накладные расходы» составят:

$$Р_Н = 3257{,}28 \cdot \frac{50\%}{100\%} = 1628{,}64 \text{ руб.}$$

Общая сумма расхода по всем статьям сметы ($С_Р$) на разработку ПС рассчитывается по формуле (5.6)

$$С_Р = З_О + З_Д + З_{СЗ} + Р_М + Р_Н. \quad (5.6)$$

Общая сумма расхода по всем статьям сметы составят:

$$С_Р = 3257{,}28 + 325{,}73 + 1239{,}72 + 244{,}40 + 1628{,}64 = 6695{,}77 \text{ руб.}$$

Затраты на сопровождение и адаптацию определяются по формуле (5.7)

$$Р_{са} = \frac{С_р \cdot Н_{рса}}{100\%}, \quad (5.7)$$

где $Н_{рса}$ – норматив расходов на сопровождение и адаптацию (20%);

$С_Р$ – смета расходов в целом по организации без расходов на сопровождение и адаптацию, руб.

Затраты на сопровождение и адаптацию составят:

$$Р_{са} = 6695{,}77 \cdot \frac{20\%}{100\%} = 1339{,}16 \text{ руб.}$$

Общая сумма расходов на разработку (с затратами на сопровождение и адаптацию) как полная себестоимость ПП (С) определяется по формуле (5.8)

$$С = С_Р + Р_{са}. \quad (5.8)$$

Полная себестоимость ПП составит:

$$С = 6695{,}77 + 1339{,}16 = 8034{,}93 \text{ руб.}$$

Таблица 5.2 – Расчеты сметы затрат и отпускной цены

Наименование статьи	Условное обозначение	Сумма, руб.
Основная заработная плата команды разработчиков	$З_О$	3257,28
Дополнительная заработная плата команды разработчиков	$З_Д$	325,73
Отчисление в ФСЗН	$З_{СЗ}$	1279,32
Машинное время	$Р_М$	244,40
Накладные расходы	$Р_Н$	1628.64

Наименование статьи	Условное обозначение	Сумма, руб.
Общая сумма расходов по всем статьям сметы	$С_Р$	6695.77
Затраты на сопровождение и адаптацию	$Р_{са}$	1339.16
Полная себестоимость	С	8034.93

Экономический эффект для организации потребителя данного программного продукта заключается в приросте прибыли за счет экономии затрат на заработную плату в связи с уменьшением трудоемкости выполнения работ по управлению трудовыми ресурсами.

Экономия затрат на заработную плату. Прирост прибыли за счет экономии расходов на заработную плату и отчислений на социальное страхование в результате снижения трудоемкости определяется по формуле:

$$С = К_{пр} * (\ t_c\ *\ T_c - \ t_н *\ \ T_н) * N_n * \left(1 + \frac{Н_д}{100\%}\right) * \ (1 + \frac{Н_{но}}{100\%}) \quad (5.9)$$

Где N_n - плановый объем работ (1800 заявок на получение и отклик на заявку клиента), шт.;

t_c, $t_н$ - трудоемкость выполнения работы до и после внедрения программного продукта, нормо-час;

Заработная плата менеджера по работе с клиентами составляет 5, 95 руб. ;

$Т_с$, $Т_н$ - часовая тарифная ставка, соответствующая разряду выполняемых работ до и после внедрения программного продукта, руб./ч;

$К_{пр}$ - коэффициент премий (1,5) ;

$Н_д$ - норматив дополнительной заработной платы (20%);

$Н_{но}$ - ставка отчислений от заработной платы, включаемых в себестоимость (34,6%);

До внедрения программного продукта составляло 0,25 чел. часа, после внедрения 0,08 чел. час;

Экономия затрат на заработную плату составит:

$$Э_з = 1{,}5 * (0{,}25 * 5{,}95 - 0{,}08 * 5{,}95) * 1800 * \left(1 + \frac{20}{100\%}\right) * \left(1 + \frac{34{,}6}{100\%}\right) = 4417{,}74 \text{ руб.}$$

Таким образом, прирост прибыли составит $Э_з = 4417{,}74$ руб.

Амортизационные отчисления являются источником погашения инвестиций в разработку программного продукта.

Расчет амортизационных отчислений осуществляется по следующей формуле:

$$А = \frac{Н_а * З}{100} \quad (5.10)$$

Где З – затраты на разработку программного продукта, руб.

$Н_а$ – норма амортизации программного продукта (25%).

$$А = \frac{8034{,}93 * 25}{100} = 2008{,}73 \text{ руб.}$$

Для расчета показателей экономической эффективности использования программного продукта необходимо полученные суммы результата (прироста чистой прибыли) и затрат (капитальных вложений) по годам приводят к единому времени – расчетному году путем умножения результатов и затрат за каждый год на коэффициент привидения ($ALFA_t$), который рассчитывается по формуле:

$$ALFA_t = (1 + E_н)^{t_р - t}, \quad (5.11)$$

где $E_н$ – норматив привидения разновременных затрат и результатов (10%);

$t_р$ – расчетный год, $t_р$ = 1;

t– номер года, результаты и затраты которого приводятся к расчетному;

2020 г. – $\alpha_1 = (1 + 0{,}1)^{1-1} = 1$;

2021 г. – $\alpha_2 = (1 + 0{,}1)^{1-2} = 0{,}909$;

2022 г. – $\alpha_3 = (1 + 0{,}1)^{1-3} = 0{,}82$;

2023 г. – $\alpha_4 = (1 + 0{,}1)^{1-4} = 0{,}751$

Результаты расчета показателей эффективности приведены в таблице 5.3.

Таблица 5.3 − Расчет экономического эффекта от использования нового ПС

Наименование показателей	По годам производства			
	2020	2021	2022	2023
Результат				
1. Экономический эффект	-	4417,74	4417,74	4417,74
2. Дисконтированный результат	-	4015,72	3622,55	3317,72
Затраты (инвестиции)				
3. Амортизационные отчисления	2008,73	2008,73	2008,73	2008,73

4. Инвестиции в разработку программного продукта	8034,93	-	-	-
5. Дисконтированеные инвестиции	8034,93	-	-	-
6. Чистый дисконтированный доход по годам	-8034,93	4015,72	3622,55	3317,72
7. ЧДД нарастающим итогом	-8034,93	-4019,21	-396,66	2921,06
Коэффициент дисконтирования	1	0,909	0,82	0,751

Рассчитаем рентабельность инвестиций в разработку и внедрение программного продукта ($Р_И$) по формуле:

$$Р_и = \frac{\sum_{t=1}^{n} P_t a_t}{\sum_{t=1}^{n} И_t a_t}, \quad (5.12)$$

где $P_t a_t$ – результат с учетом фактора времени, руб.;

$И_t a_t$ – инвестиции с учетом фактора, руб.;

n – количество лет в расчетном периоде, равное 4.

Тогда рентабельность составит:

$Р_и$=(4015,72+3622,55+3317,72)/ 8034,93 = 136%

В результате технико-экономического обоснования применения программного продукта были получены следующие значения показателей их эффективности: чистый дисконтированный доход за четыре года работы программы составит 2921,06 руб.; затраты на разработку программного продукта окупятся на четвертый год его использования; рентабельность инвестиций составляет 136%.

Таким образом, применение программного продукта является эффективным и инвестиции в его разработку целесообразно осуществлять.

6 ВОПРОСЫ ОХРАНЫ ТРУДА ПРИ РАБОТЕ С КОМПЬЮТЕРАМИ

6.1 Идентификация и анализ вредных и опасных факторов в проектируемом объекте

Вредные факторы, воздействующие на человека при работе с компьютерной техникой, классифицируемых в соответствии с государственным стандартом ГОСТ 12.0.003-74 ССБТ. "Опасные и вредные производственные факторы. Классификация". К опасным и вредным производственным факторам относятся:

- повышенные уровни электромагнитного излучения. Источниками излучения могут быть системный блок, беспроводные системы передачи информации на расстояния, клавиатура, мышь [17]. Действие электромагнитных полей на организм человека в основном определяется поглощенной в нем энергией. Эта энергия преобразуется в тепло, в результате чего наступает опасный перегрев. Кроме того, электромагнитные поля вызывают нарушения в работе сердечно-сосудистой и нервной систем.

- повышенные уровни ультрафиолетового излучения. Источником излучения является монитор. Повышенные дозы ультрафиолетового излучения могут вызывать головную боль, головокружение, повышение температуры тела, повышение утомляемости и др.

- повышенные уровни инфракрасного излучения. Инфракрасное излучение негативно влияет на зрительные органы человека. Может вызвать нарушения в работе сетчатки, хрусталика, радужной оболочки глаза. Инфракрасное излучение коротковолнового диапазона может воздействовать на внутренние органы, мозговую ткань, а при воздействии на глаза может вызвать появление инфракрасной катаракты.

- повышенные уровни статического электричества. При воздействии повышенного уровня статического электричества возможно появление головной боли, нарушение сна, снижение аппетита и др. Вредное воздействие на организм человека статическое электричество оказывает не только при непосредственном его контакте с зарядом, но и за счет действия электрического поля, возникающего вокруг заряженных поверхностей. В вычислительных центрах разрядные токи статического электричества чаще всего возникают при прикосновении к любому из элементов компьютера. Такие разряды могут привести к выходу из строя техники.

- пониженная или повышенная влажность воздуха рабочей зоны. При относительной влажности более 75-80 % снижается сопротивление изоляции, изменяются рабочие характеристики, и повышается интенсивность отказов элементов ПК. Повышенная влажность

воздуха приводит к нарушению терморегуляции, уменьшается испарение пота, что приводит к резкому ухудшению состояния здоровья человека и снижению его работоспособности. Понижение влажности воздуха вызывает неприятные ощущения сухости слизистых оболочек дыхательных путей.

- пониженная или повышенная подвижность воздуха рабочей зоны. Скорость движения воздуха оказывает влияние на распределение вредных частиц в помещении и влияет на отдачу теплоты организмом. Повышенная скорость движения воздуха, особенно в условиях низких температур, вызывает увеличение теплопотерь и ведет к сильному охлаждению организма.

- пониженная или повышенная температура воздуха рабочей зоны. Повышенная температура приводит к быстрой утомляемости, повышению артериального давления, ослаблению внимания и др. При пониженной температуре организм человека переохлаждается, что ведет к простудным заболеваниям.

- повышенный уровень шума. Среди многочисленных проявлений воздействия шума на организм можно выделить снижение разборчивости речи, неприятные ощущения, развитие утомления, снижение производительности труда и возникновение шумовой патологии. При действии интенсивного шума изменения со стороны нервной системы значительно более выражены и предшествуют развитию патологии органа слуха. У рабочих преобладают жалобы на головные боли, несистематические головокружения, снижение памяти, повышенную утомляемость, эмоциональную неустойчивость, нарушение сна, сердцебиение и боли в области сердца, снижение аппетита и др. Источника-ми шума на рабочем месте являются: вентиляторы систем охлаждения, накопители, монитор, клавиатура, принтер, преобразователи напряжения.

- повышенный или пониженный уровень освещенности. Недостаточное и чрезмерное освещение рабочего места оказывает отрицательное воздействие на зрение человека, вызывает эмоциональное напряжение, что ведет к снижению производительности труда.

- повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека. Поражение электрическим током опасно для здоровья и жизни человека и обусловлено тем, что проходящий ток не видим человеком и зачастую не воспринимается как источник опасности. Степень поражения человека электрическим током зависит от схемы включения человека в сеть, степени изоляции токоведущих частей, напряжения в сети, схемы сети, индивидуальных свойств человека. Проходя через организм, электрический ток оказывает термическое, электролитическое и биологическое воздействие. Это многообразие действий электрического тока приводит к различным травмам от легкого пощипывания до остановки сердца.

- статические перегрузки костно-мышечного аппарата и динамические локальные перегрузки мышц кистей рук. Основной причиной перенапряжения мышц спины и ног являются нерациональная высота рабочей поверхности стола и сидения, отсутствие опорной спинки и подлокотников, неудобное размещение монитора и клавиатуры, отсутствие подставки для ног. Длительная монотонная работа с клавиатурой может привести к развитию воспалительных процессов тканей сухожилий.

- умственное перенапряжение и эмоциональные перегрузки. Работа за компьютером требует большой концентрации и внимания. Последствиями воздействия такого рода нагрузок приводят к быстрому переутомлению, головным болям, плохому сну и снижению работоспособности.

- монотонность труда. Может вызвать быстрое развитие усталости в связи с локализацией мышечных и нервных нагрузок, гиподинамию, развитие неврозов, недовольство работой и снижение творческой активности.

- пожароопасность. Источниками возгорания могут быть электронные схемы, приборы, применяемые для технического обслуживания, устройства электропитания, кондиционирования воздуха, где в результате различных нарушений образуются перегретые элементы, электрические искры и дуги. Для отвода избыточной теплоты служат системы вентиляции и кондиционирования воздуха. Но при постоянном действии эти системы представляют собой дополнительную пожарную опасность. Опасными для человека факторами пожара являются: огонь (ожоги кожи), токсичные продукты горения, повышенная температура среды (вызывают потерю сознания, поражение и некроз верхних дыхательных путей), дым (вызывает кашель, отек легких), пониженная концентрация кислорода (вызывает ухудшение двигательной функции организма), взрывы, вытекание опасных веществ (вызывает ранения обломками, химические ожоги и отравления), разрушения строительных конструкций, паника.

6.2 Технические, технологические, организационные решения по устранению опасных и вредных факторов, разработка защитных средств.

Параметры факторов производственной среды на рабочих местах с использованием ЭВМ регламентируются в СаНПиН, утвержденных Постановлением Минздрава РБ от 28.06.2013 № 59 «Требования при работе с видеодисплейными терминалами и электронно-вычислительными машинами» и гигиеническом нормативе «Предельно допустимые уровни нормируемых параметров при работе с видеодисплейными терминалами (ВДТ) и электронно-вычислительными машинами»[18].

Для работы используется жидкокристаллический монитор, что обеспечивает защиту от инфракрасного, ультрафиолетового, ионизирующего и электромагнитного излучения. Используемые мониторы соответствуют СанПиН, т.к. их напряженность электромагнитного поля по электрической составляющей не превышает 25 В/м. Интенсивность инфракрасного и видимого излучения от экрана монитора не превышает 0,1 Вт/м2 в видимом (400 – 760 нм) диапазоне. Для защиты от электромагнитного излучения устанавливается рациональное время работы оборудования и работников.

С целью снижения уровня статического электричества пол в помещении покрыт антистатическим материалом.

Показатели микроклимата на рабочем месте соответствуют требованиям СаНПиН, утвержденным Постановлением минздрава РБ от 28.06.2013 №59 «Требования при работе с видеодисплейными терминалами и электронно-вычислительными машинами». Для обеспечения нормативных параметров микроклимата используются как организационные методы (рациональная организация проведения работ в зависимости от времени года и суток, чередование труда и отдыха), так и технические средства (вентиляция, кондиционирование воздуха, отопительная система).

В помещении используется совмещенное освещение. Искусственное освещение выполняется в виде общего освещения люминесцентными лампами. Светильники располагаются над рабочей поверхностью в равномерно-прямоугольном порядке. Такой вариант освещения обеспечивает необходимый уровень освещенности (300лк), не оказывает слепящего действия на оператора. Для предотвращения образования бликов на экране, применяются светильники общего освещения, при этом линии светильников располагаются параллельно светопроемам.

Электробезопасность в проектируемом объекте обеспечивается в соответствии с ГОСТ 12.1.019-2017 ССБТ. «Электробезопасность. Общие требования и номенклатура видов защиты». В качестве основного способа защиты используется защитное заземление.

Для того чтобы снизить риск переутомления, истощения нервно-психических ресурсов организма, нужно соблюдать регламентированные перерывы в работе.

Для обнаружения начальной стадии возгорания и оповещения службы пожарной охраны используется система автоматической пожарной сигнализации. В качестве основного средства пожаротушения используются порошковые огнетушители ОП-4, которые позволяют тушить электроустановки.

7 ЭНЕРГО-И РЕСУРСОСБЕРЕЖЕНИЕ

На развитие предприятий в нашей стране существенное негативное влияние оказывает высокая доля энергетических затрат, которая составляет в среднем 8-12% и имеет устойчивую тенденцию к росту в связи с большим моральным и физическим износом основного оборудования и значительными потерями при транспортировке энергетических ресурсов.

Одним из определяющих условий снижения издержек на промышленных предприятиях и повышения экономической эффективности производства в целом является рациональное использование энергетических ресурсов. Вместе с тем, энергосберегающий путь развития отечественной экономики возможен только при формировании и последующей реализации программ энергосбережения на отдельных предприятиях, для чего необходимо создание соответствующей методологической и методической базы. Откладывание реализации энергосберегающих мероприятий наносит значительный экономический ущерб предприятиям и негативно отражается на общей экологической и социально-экономической ситуации. Помимо этого, дальнейший рост издержек в промышленности и других отраслях народного хозяйства сопровождается растущим дефицитом финансовых ресурсов, что задерживает обновление производственной базы предприятий в соответствии с достижениями научно-технического прогресса.

Для предотвращения финансовых потерь при формировании совокупности энергосберегающих мероприятий требуется разработка и совершенствование методов оценки эффективности программ энергосбережения, учитывающих многовариантность использования источников инвестиций, предназначенных для их реализации. Уменьшение энергетической составляющей в издержках производства позволит получить дополнительные средства для обеспечения приемлемого уровня морального и физического износа технологического оборудования.

При этом необходимо отметить, что в качестве ориентира энергосбережения могут применяться различные критерии. Наиболее часто ориентиром для управляющих воздействий служит потенциал энергосбережения, под которым подразумевают резервы, которые могут быть освоены во времени. Проводя анализ и оценку экономического энергоресурсного потенциала необходимо рассматривать не только количественную и качественную его характеристики, но и возможность рационального использования энергетических ресурсов.

ЗАКЛЮЧЕНИЕ

В данной работе представлена разработка программного средства целевого управления бизнес-процессами предприятия сферы услуг, которая обладает следующими функциональными возможностями для реализации задач внутреннего взаимодействия:

− имеет интуитивно понятный интерфейс для осуществления навигации по приложению;

− осуществляет вход в приложение с использованием собственных данных для аутентификации;

− создает информационные потоки для каждой группы пользователей;

− осуществляет уведомление пользователей

− имеет возможность менеджеру выбирать рабочего специалиста на заказ из списка откликнувшихся специалистов

− имеет возможность редактирования личных данных

− добавляет новые типы услуг при необходимости

Для разработки справочно-поисковой системы был выбран следующий комплекс средств:

− комплекс инструментальных средств IntelliJ IDEA для разработки серверной и клиентской частей приложения;

− фреймворк Sprong Framework как средство разработки web-приложений;

− СУБД Microsoft SQL Server как средство хранение данных.

К преимуществам разработанной системы управления бизнес-процессами следует отнести:

− адаптированный под нужды сотрудников интерфейс;

− единую систему авторизации;

− доступ на основе ролей;

− гибкую адаптацию под новые требования, возможность добавления новых сервисов и функций;

− высокую доступность приложения;

− осуществление обмена информацией без поддержки IT-персонала.

Таким образом, система, обладая перечисленными преимуществами, в полной мере сможет восполнить потребность сотрудников компании в эффективном инструменте для информационного обмена и внутреннего взаимодействия.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Структурный (процессный) подход к моделированию бизнес-процессов [Электронный ресурс]. – Режим доступа: https://wiki.clan.su/publ/strukturnyj_processnyj_podkhod_k_modelirovaniju_biznes_processov/1-1-0-28. – Дата доступа: 22.04.2021.
2. Сбалансированная система показателей [Электронный ресурс]. – Режим доступа: https://www.businessstudio.ru/articles/article/bsc_sbalansirovannaya_sistema_pokazateley_i_busine/. – Дата доступа: 22.04.2021.
3. Салон красоты TIFFANY [Электронный ресурс]. – Режим доступа: https://tiffany.by. – Дата доступа: 22.04.2021.
4. Студии красоты "Blond&Brown" [Электронный ресурс]. – Режим доступа: http://blonda.by/about. – Дата доступа: 22.04.2021.
5. Гостиница красоты «Шуры-Муры» [Электронный ресурс]. – Режим доступа: https://shury-mury.by. – Дата доступа: 22.04.2021.
6. Студия красоты Аделины Маффей [Электронный ресурс]. – Режим доступа: http://amstudia.by. – Дата доступа: 22.04.2021.
7. Студия красоты «BellaNika (Бэлла Ника)» [Электронный ресурс]. – Режим доступа: https://bellanika.103.by. – Дата доступа: 22.04.2021.
8. Язык программирования Java [Электронный ресурс]. – Режим доступа: https://www.internet-technologies.ru/articles/kak-nauchitsya-programmirovat-na-java.html. – Дата доступа: 22.04.2021.
9. Что такое Spring? Обзор фреймворка Java [Электронный ресурс]. – Режим доступа: https://oracle-patches.com/coding/3935-chto-takoe-spring-obzor-frejmvorka-java. – Дата доступа: 22.04.2021.
10. Thymeleaf [Электронный ресурс]. – Режим доступа: https://www.thymeleaf.org. – Дата доступа: 22.04.2021.
11. HTML [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/HTML. – Дата доступа: 22.04.2021.
12. Основы CSS [Электронный ресурс]. – Режим доступа: https://html5book.ru/osnovy-css/. – Дата доступа: 22.04.2021.
13. Современный учебник JavaScript [Электронный ресурс]. – Режим доступа: https://learn.javascript.ru. – Дата доступа: 22.04.2021.

14. Spring MVC — основные принципы [Электронный ресурс]. – Режим доступа: https://habr.com/ru/post/336816/. – Дата доступа: 22.04.2021.

15. Диаграмма последовательностей (sequence diagram) [Электронный ресурс]. – Режим доступа: https://intuit.ru/studies/courses/1007/229/lecture/5954?page=3. – Дата доступа: 22.04.2021.

16. UML — диаграмма вариантов использования [Электронный ресурс]. – Режим доступа: https://habr.com/ru/post/47940/. – Дата доступа: 22.04.2021.

17. ГОСТ 12.0.003-74 Система стандартов безопасности труда (ССБТ). Опасные и вредные производственные факторы. Классификация (с Изменением N 1) [Электронный ресурс]. – Режим доступа: http://docs.cntd.ru/document/5200224. – Дата доступа: 22.04.2021.

18. ПОСТАНОВЛЕНИЕ МИНИСТЕРСТВА ЗДРАВООХРАНЕНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ [Электронный ресурс]. – Режим доступа: http://ripo.unibel.by/assets/ripo_new/docs/ohr_truda.pdf. – Дата доступа: 22.04.2021.